

METHOD AND SYSTEM FOR TRACKING MULTIPLE REGIONAL  
OBJECTS BY MULTI-DIMENSIONAL RELAXATION

## CROSS-REFERENCE TO RELATED APPLICATIONS

5        This application is a continuation-in-part of U.S. Patent  
Application Serial No. 08/404,024, filed March 14, 1995 and  
issued July 16, 1996 as U.S. Patent No. 5,537,119, which is a  
continuation-in-part of U.S. Patent Application Serial No.  
08/171,327 filed December 21, 1993, now U.S. Patent No.  
10 5,406,289.

## BACKGROUND OF THE INVENTION

a. Field of Invention

15        The invention relates generally to computerized  
techniques for processing data obtained from radar to track  
multiple discrete objects.

b. Description of the Background

20        There are many situations where the courses of multiple  
objects in a region must be tracked. Typically, radar is used  
to scan the region and generate discrete images or "snapshots"  
based on sets of returns or observations. In some types of  
tracking systems, all the returns from any one object are  
represented in an image as a single point unrelated to the  
shape or size of the objects. "Tracking" is the process of  
25 identifying a sequence of points from a respective sequence of

the images that represents the motion of an object. The tracking problem is difficult when there are multiple closely spaced objects because the objects can change speed and direction rapidly and move into and out of the line of sight for other objects. The problem is exacerbated because each set of returns may result from noise as well as echoes from the actual objects. The returns resulting from the noise are also called false positives. Likewise, the radar will not detect all echoes from the actual objects and this phenomena is called a false negative or "missed detect" error. For tracking airborne objects, a large distance between the radar and the objects diminishes the signal to noise ratio so the number of false positives and false negatives can be high. For robotic applications, the power of the radar is low and as a result, the signal to noise ratio can also be low and the number of false positives and false negatives high.

In view of the proximity of the objects to one another, varied motion of the objects and false positives and false negatives, multiple sequential images should be analyzed collectively to obtain enough information to properly assign the points to the proper tracks. Naturally, the larger the number of images that are analyzed, the greater the amount of information that must be processed.

While identifying the track of an object, a kinematic model may be generated describing the location, velocity and acceleration of the object. Such a model provides the means by which the future motion of the object can be predicted.

- 5 Based upon such a prediction, appropriate action may be initiated. For example, in a military application there is a need to track multiple enemy aircraft or missiles in a region to predict their objective, plan responses and intercept them. Alternatively, in a commercial air traffic control application
- 10 there is a need to track multiple commercial aircraft around an airport to predict their future courses and avoid collision. Further, these and other applications, such as robotic applications, may use radar, sonar, infrared or other object detecting radiation bandwidths for tracking objects.
- 15 In particular, in robotic applications reflected radiation can be used to track a single object which moves relative to the robot (or vice versa) so the robot can work on the object.

Consider the very simple example of two objects being tracked and no false positives or false negatives. The radar,

20 after scanning at time  $t_1$ , reports objects at two locations in a first observation set. That is, the radar returns a set of two observations  $\{o_{11}, o_{12}\}$ . At time  $t_2$  the radar returns a similar set of two observations  $\{o_{21}, o_{22}\}$  from a second

observation set. Suppose from prior processing that track data for two tracks  $T_1$  and  $T_2$  includes the locations at  $t_0$  of two objects. Track  $T_1$  may be extended through the points in the two sets of observations in any of four ways, as may track  $T_2$ . The possible extensions of  $T_1$  can be described as:  $\{T_1, o_{11}, o_{21}\}$ ,  $\{T_1, o_{11}, o_{22}\}$ ,  $\{T_1, o_{12}, o_{21}\}$  and  $\{T_1, o_{12}, o_{22}\}$ . Tracks can likewise be extended from  $T_2$  in four possible ways, including  $\{T_2, o_{12}, o_{21}\}$ . Figure 1 illustrates these five (out of eight) possible tracks (to simplify the problem for purposes of explanation). The five track extensions are labeled  $h_{11}$ ,  $h_{12}$ ,  $h_{13}$ ,  $h_{14}$ , and  $h_{21}$  wherein  $h_{11}$  is derived from  $\{T_1, o_{11}, o_{21}\}$ ,  $h_{12}$  is derived from  $\{T_1, o_{11}, o_{22}\}$ ,  $h_{13}$  is derived from  $\{T_1, o_{12}, o_{21}\}$ ,  $h_{14}$  is derived from  $\{T_1, o_{12}, o_{22}\}$ , and  $h_{21}$  is derived from  $\{T_2, o_{11}, o_{21}\}$ . The problem of determining which such track extensions are the most likely or optimal is hereinafter known as the assignment problem.

It is known from prior art to determine a figure of merit or cost for assigning each of the points in the images to a track. The figure of merit or cost is based on the likelihood that the point is actually part of the track. For example, the figure of merit or cost may be based on the distance from the point to an extrapolation of the track. Figure 1 illustrates costs  $\delta_{21}$  and  $\delta_{22}$  for hypothetical extension  $h_{21}$  and

modeled target characteristics. The function to calculate the cost will normally incorporate detailed characteristics of the sensor, such as probability of measurement error, and track characteristics, such as likelihood of track maneuver.

5        Figure 2 illustrates a two by two by two matrix,  $\mathbf{c}$ , that contains the costs for each point in relation to each possible track. The cost matrix is indexed along one axis by the track number, along another axis by the image number and along the third axis by a point number. Thus, each position in the cost  
10        matrix lists the cost for a unique combination of points and a track, one point from each image. Figure 2 also illustrates a  $\{0, 1\}$  assignment matrix,  $\mathbf{z}$ , which is defined with the same dimensions as the cost matrix. Setting a position in the assignment matrix to "one" means that the equivalent position  
15        in the cost matrix is selected into the solution. The illustrated solution matrix selects the  $\{h_{14}, h_{21}\}$  solution previously described. Note that for the above example of two tracks and two snapshots, the resulting cost and assignment matrices are three-dimensional. As used in this patent  
20        application, the term "dimension" means the number of axes in the cost or assignment matrix while size refers to the number of elements along a typical axis. The costs and assignments have been grouped in matrices to facilitate computation.

A solution to the assignment problem satisfies two constraints - first, the sum of the associated costs for assigning points to a track extension is minimized and, second, if no false positives or false negatives exist, then  
5 each point is assigned to one and only one track.

When false positives exist, however, additional hypothetical track extensions incorporating the false positives will be generated. Further note that the random locations of false positives will, in general, not fit well  
10 with true data and such additional hypothetical track extensions will result in higher costs. Also note that when false negative errors exist, then the size of the cost matrix must grow to include hypothetical track extensions formulated with "gaps" (i.e., data omissions where there should be  
15 legitimate observation data) for the false negatives. Thus, the second criteria must be weakened to reflect false positives not being as signed and also to permit the gap filler to be multiply assigned. With hypothetical cost calculated in this manner then the foregoing criteria for  
20 minimization will tend to materialize the false negatives and avoid the false positives.

For a three-dimensional problem, as is illustrated in Figure 1, but with  $N_1$  (initial) tracks,  $N_2$  observations in scan

1,  $N_3$  observations in scan 2, false positives and negatives assumed, the assignment problem can be formulated as:

$$\begin{aligned}
 (a) \quad & \text{Minimize:} \quad \sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} \sum_{i_3=0}^{N_3} c_{i_1 i_2 i_3} z_{i_1 i_2 i_3} \\
 (b) \quad & \text{Subject to:} \quad \sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} = 1, \quad i_1 = 1, \dots, N_1, \quad (0.1) \\
 (c) \quad & \sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} \leq 1, \quad i_2 = 1, \dots, N_2, \\
 (d) \quad & \sum_{i_1=1}^{N_1} \sum_{i_2=1}^{N_2} z_{i_1 i_2 i_3} \leq 1, \quad i_3 = 1, \dots, N_3, \\
 (e) \quad & z_{i_1 i_2 i_3} \in \{0, 1\} \quad \forall \quad z_{i_1 i_2 i_3},
 \end{aligned}$$

where "c" is the cost and "z" is a point or observation assignment, as in Figure 2.

The minimization equation or equivalently objective function (0.1)(a) specifies the sum of the element by element product of the **c** and **z** matrices. The summation includes hypothesis representations  $z_{i_1 i_2 i_3}$  with observation number zero being the gap filler observation. Equation (0.1)(b) requires that each of the tracks  $T_1, \dots, T_{N_1}$  be extended by one and only one hypothesis. Equation (0.1)(c) relates to each point or observation in the first observation set and requires that each such observation, except the gap filler, can only

associate with one track but, because of the "less than" condition, it might not associate with any track. Equation (0.1)(d) is like (0.1)(c) except that it is applicable to the second observation set. Equation (0.1)(e) requires that  
5 elements of the solution matrix  $z$  be limited to the zero and one values.

The only known method to solve Problem Formulation (0.1) exactly is a method called "Branch and Bound." This method provides a systematic ordering of the potential solutions so  
10 that solutions with a same partial solution are accessible via a branch of a tree describing all possible solutions whereby the cost of unexamined solutions on a branch are incrementally developed as the cost for other solutions on the branch are determined. When the developing cost grows to exceed the  
15 previously known minimal cost (i.e., the bound) then enumeration of the tree branch terminates. Evaluation continues with a new branch. If evaluation of the cost of a particular branch completes, then that branch has lower cost than the previous bound so the new cost replaces the old  
20 bound. When all possible branches are evaluated or eliminated then the branch that had resulted in the last used bound is the solution. If we assume that  $N_1 = N_2 = N_3 = n$  and that branches typically evaluate to half there full length, then



workload associated with "branch and bound" is proportional to  $(n!|\frac{n}{2}|)^2$ . This workload is unsuited to real time evaluation.

The Branch and Bound algorithm has been used in past research on the Traveling Salesman Problem. Messrs. Held and  
5 Karp showed that if the set of constraints was relaxed by a method of Lagrangian Multipliers (described in more detail below) then tight lower bounds could be developed in advance of enumerating any branch of the potential solution. By  
10 initiating the branch and bound algorithm with such a tight lower bound, significant performance improvements result in that branches will typically evaluate to less than half their full length.

Messrs. Frieze and Yadagar in dealing with a problem related to scheduling combinations of resources, as in job,  
15 worker and work site, showed that Problem Formulation (0.1) applied. They further described a solution method based upon an extension of the Lagrangian Relaxation previously mentioned. The two critical extensions provided by Messrs. Frieze and Yadagar were: (1) an iterative procedure that  
20 permitted the lower bound on the solution to be improved (by "hill climbing" described below) and (2) the recognition that when the lower bound of the relaxed problem was maximized, then there existed a method to recover the solution of the

non-relaxed problem in most cases using parameters determined at the maximum. The procedures attributed to Messrs. Frieze and Yadagar are only applicable to the three-dimensional problem posed by Problem Formulation (0.1) and where the cost  
5 matrix is fully populated. However, tracking multiple airborne objects usually requires solution of a much higher dimensional problem.

Figures 1 and 2 illustrate an example where "look ahead" data from the second image improved the assignment accuracy  
10 for the first image. Without the look ahead, and based only upon a simple nearest neighbor approach, the assignments in the first set would have been reversed. Problem Formulation (0.1) and the prior art only permit looking ahead one image. In the prior art it was known that the accuracy of assignments  
15 will improve if the process looks further ahead, however no practical method to optimally incorporate look ahead data existed. Many real radar tracking problems involve hundreds of tracks, thousands of observations per observation set and matrices with dimensions in the range of 3 to 25 including  
20 many images of look ahead.

It was also known that the data assignment problem may be simplified (without reducing the dimension of the assignment problem) by eliminating from consideration for each track

those points which, after considering estimated limits of speed and turning ability of the objects, could not physically be part of the track. One such technique, denoted hereinafter the "cone method," defines a cone as a continuation of each  
5 previously determined track with the apex of the cone at the end of the previously defined track. The length of the cone is based on the estimated maximum speed of the object and the size of the arc of the cone is based on the estimated maximum turning ability of the object. Thus, the cone defines a  
10 region outside of which no point could physically be part of the respective track. For any such points outside of the cones, an infinite number could be put in the cost matrix and a zero could be preassigned in the assignment matrix. It was known for the tracking problem that these elements will be  
15 very common in the cost and selection matrices (so these matrices are "sparse").

It was also known in the prior art that one or more tracks which are substantially separated geographically from the other tracks can be separated also in the assignment  
20 problem. This is done by examining the distances from each point to the various possible tracks. If the distances from one set of points are reasonably short only in relation to one track, then they are assigned to that track and not further

considered with the remainder of the points. Similarly, if a larger group of points can only be assigned to a few tracks, then the group is considered a different assignment problem. Because the complexity of assignment problems increases  
5 dramatically with the number of possible tracks and the total number of points in each matrix, this partitioning of the group of points into a separate assignment problem and removal of these points from the matrices for the remaining points, substantially reduces the complexity of the overall assignment  
10 problem.

A previously known Multiple Hypothesis Testing (MHT) algorithm (see Chapter 10 of S.S. Blackman. Multiple-Target Tracking with Radar Applications. Artech House, Norwood, MA, 1986.) related to formulation and scoring. The MHT procedure  
15 describes how to formulate the sparse set of all reasonable extension hypothesis (for Figure 1 the set  $\{h_{11} \dots h_{24}\}$ ) and how to calculate a cost of the hypothesis  $\{T_i, o_{1j}, o_{2k}\}$  based upon the previously calculated cost for hypothesis  $\{T_i, o_{1j}\}$ . The experience with the MHT algorithm, known in the prior art, is  
20 the basis for the assertion that look ahead through  $k$  sets of observations results in improved assignment of observations from the first set to the track.

In theory, the MHT procedure uses the extendable costing procedure to defer assignment decision until the accumulated evidence supporting the assignment becomes overwhelming. When it makes the assignment decision, it then eliminates all potential assignments invalidated by the decision in a process called "pruning the tree." In practice, the MHT hypothesis tree is limited to a fixed number of generations and the overwhelming evidence rule is replaced by a most likely and feasible rule. This rule considers each track independently of others and is therefore a local decision rule.

A general object of the present invention is to provide an efficient and accurate process for assigning each point object in a region from multiple images to a proper track and then taking an action based upon the assignments.

A more specific object of the present invention is to provide a technique of the foregoing type which determines the solution of a  $k$ -dimensional assignment problem where " $k$ " is greater than or equal to three.

## SUMMARY OF THE INVENTION

The present invention relates to a method and apparatus for tracking objects. In particular, the present invention tracks movement or trajectories of objects by analyzing

radiation reflected from the objects, the invention being especially useful for real-time tracking in noisy environments.

In providing such a tracking capability, a region  
5 containing the objects is repeatedly scanned to generate a multiplicity of sequential images or data observation sets of the region. One or more points (or equivalently observations), in each of the images or observation sets are detected wherein each such observation either corresponds to  
10 an actual location of an object or is an erroneous data point due to noise. Subsequently, for each observation detected, figures of merit or costs are determined for assigning the observation to each of a plurality of previously determined tracks. \*Afterwards, a first optimization problem is  
15 specified which includes:

- (a) a first objective function for relating the above mentioned costs to potential track extensions through the detected observations (or simply observations); and
- (b) a first collection of constraint sets wherein each  
20 constraint set includes constraints to be satisfied by the observations in a particular scan to which the constraint set is related. In general, there is a constraint for each observation of the scan, wherein the

constraint indicates the number of track extensions to which the observation may belong.

In particular, the first optimization problem is formulated, generated or defined as an  $M$ -dimensional assignment problem wherein there are  $M$  constraint sets in the first collection of constraint sets (i.e., there are  $M$  scans being examined) and the first objective function minimizes a total cost for assigning observations to various track extensions wherein terms are included in the cost, such that the terms have the figures of merit or costs for hypothesized combinations of assignments of the observations to the tracks. Subsequently, the formulated  $M$ -dimensional assignment problem is solved by reducing the complexity of the problem by generating one or more optimization problems each having a lower dimension and then solving each lower dimension optimization problem. That is, the  $M$ -dimensional assignment problem is solved by solving a plurality of optimization problems each having a lower number of constraint sets.

The reduction of the  $M$ -dimensional assignment problem to a lower dimensioned problem is accomplished by relaxing the constraints on the points of one or more scans thereby permitting these points to be assigned to more than one track extension. In relaxing the constraints, terms having penalty

factors are added into the objective function thereby increasing the total cost of an assignment when one or more points are assigned to more than one track. Thus, the reduction in complexity by this relaxation process is  
 5 iteratively repeated until a sufficiently low dimension is attained such that the lower dimensional problem may be solved directly by known techniques.

In one embodiment of the invention, each  $k$ -dimensional assignment problem ( $2 < k \leq M$ ) is iteratively reduced to a ( $k$ -  
 10  $1$ )- dimensional problem until a two-dimensional problem is specified or formulated. Subsequently, the two-dimensional problem formulated is solved directly and a "recovery" technique is used to iteratively recover an optimal or near-optimal solution to each  $k$ -dimensional problem from a derived  
 15  $(k-1)$ -dimensional problem  $k= 2,3,4,\dots M$ .

In performing each recovery step (of obtaining a solution to a  $k$ -dimensional problem using a solution to a  $(k-1)$ -dimensional problem), an auxiliary function is utilized. In particular, to recover an optimal or near-optimal solution to  
 20 a  $k$ -dimensional problem, an auxiliary function,  $\psi_{k-1}$  ( $k=4,5,\dots,M$ ), is specified and a region or domain is determined wherein this function is maximized, whereby values of the region determine the penalty factors of the  $(k-1)$ -



dimensional problem such that another two-dimensional problem can be formulated which determines a solution to the  $k$ -dimensional problem using the penalty factors of the  $(k-1)$ -dimensional problem.

5        Each Auxiliary function  $\psi_k$ ,  $k=3, \dots, M-1$ , is a function of both lower dimensional problems, penalty factors, and a solution at the dimension  $k$  at which the penalized cost function is solved directly (typically a two-dimensional problem). Further, in determining, for auxiliary function  $\psi_k$ ,  
 10    a peak region, a gradient of the auxiliary function is determined, and utilized to identify the peak region. Thus, gradients are used for each of the approximation functions  $\psi_3$ ,  $\psi_4, \dots, \psi_{M-1}$  which are sequentially formulated and used in determining penalty factors until  $M-1$  is used in determining  
 15    the penalty factors for the  $(M-1)$ -dimensional problem. Subsequently, once the  $M$ -dimensional problem is solved (using a two-dimensional problem to go from an  $(M-1)$ -dimensional solution to an  $M$ -dimensional solution), one or more of the following actions are taken based on the track assignments:  
 20    sending a warning to aircraft or a ground or sea facility, controlling air traffic, controlling anti-aircraft or anti-missile equipment, taking evasive action, working on one of the objects.

According to one feature of this first embodiment of the present invention, the following steps are also performed before the step of defining the auxiliary function. A preliminary auxiliary function is defined for each of the lower dimensional problems having a dimension equal or one greater than the dimension at which the penalized cost function is solved directly. The preliminary auxiliary function is a function of lower order penalty values and a solution at the dimension at which the penalized cost function was solved directly. In determining a gradient of the preliminary auxiliary function, step in the direction of the gradient to identify a peak region of the preliminary auxiliary function and determine penalty factors at the peak region. Iteratively repeat the defining, gradient determining, stepping and peak determining steps to define auxiliary functions at successively higher dimensions until the auxiliary function dimension  $(k-1)$  is determined. In an alternative second embodiment of the present invention, instead of reducing the dimensionality of the  $M$ -dimensional assignment problem by a single dimension at a time, a plurality of dimensions are relaxed simultaneously. This new strategy has the advantage that when the  $M$ -dimensional problem is relaxed directly to a two-dimensional assignment problem,

then all computations may be performed precisely without utilizing an auxiliary function such as  $\psi_k$  as in the first embodiment. More particularly, the second embodiment solves the first optimization problem (i.e., the  $M$ -dimensional assignment problem) by specifying (i.e., creating, generating, 5 formulating and/or defining) a second optimization problem. The second optimization problem includes a second objective function and a second collection of constraint sets wherein:

- a) the second objective function is a combination of the 10 first objective function and penalty factors or terms determined for incorporating  $(M-m)$ -constraint sets of the first optimization problem into the second objective function;
- b) the constraint sets of the second collection include only 15  $m$  of the constraint sets of the first collection of constraints,

wherein  $2 \leq m \leq M - 1$ . Note that, once the second optimization problem has been specified or formulated, an optimal or near-optimal solution is determined and that solution is used in 20 specifying (i.e., creating, generating, formulating and/or defining) a third optimization problem of  $(M-m)$  dimensions (or equivalently constraint sets). The third optimization problem is subsequently solved by decomposing it using the same

procedure of this second embodiment as was used to decompose the first optimization problem above. Thus, a plurality of instantiations of the third optimization problem are specified, each successive instantiation having a lower number of dimensions, until an instance of the third optimization problem is a two-dimensional assignment problem which can be solved directly. Subsequently, whenever an instance of the third optimization problem is solved, the solution is used to recover a solution to the instance of the first optimization problem from which this instance of the third optimization was derived. Thus, an optimal or near-optimal solution to the original first optimization problem may be recovered through iteration of the above steps.

As mentioned above, the second embodiment of the present invention is especially advantageous when  $m = 2$ , since in this case all computations may be performed precisely and without utilizing auxiliary functions.

Other features and benefits of the present invention will become apparent from the detailed description with the accompanying drawings contained hereinafter.

## BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a graph of images or data sets generated by a scan of a region and possible tracks within the images or data sets according to the prior art.

5        Figure 2 illustrates cost and assignment matrices for the data sets of Figure 1 according to the prior art.

Figure 3 is a block diagram of the present invention.

Figure 4 is a flow chart of a process according to the prior art for solving a three-dimensional assignment problem.

10       Figure 5 is a flow chart of a process according to the present invention for solving a  $k$ -dimensional assignment problem where " $k$ " is greater than or equal to 3.

Figure 6 is a graph of various functions used to explain the present invention.

15       Figure 7 is another block diagram of the present invention for solving the  $k$ -dimensional assignment problem where " $k$ " is greater than or equal to three (3).

Figure 8 is a flowchart describing the procedure for solving an  $n$ -dimensional assignment problem according to the  
20    second embodiment of the invention.

## DETAILED DESCRIPTION OF THE INVENTION

Referring now to the other figures in detail wherein like reference numerals indicate like elements throughout the several views, Figure 3 illustrates a system generally designated 100 for implementing the present invention. System 100 comprises, for example, a radar station 102 (note sonar, microwave, infrared and other radiation bandwidths are also contemplated) for scanning a region which may be, for example, an aerial region (in aerial surveillance applications) or a work region (in robotic applications) and generating signals indicating locations of objects within the region. The signals are input to a converter 104 which converts the signals to data points or observations in which each object (or false positive) is represented by a single point. The output of the converter is input to and readable by a computer 106. As described in more detail below, the computer 106 assigns the points to respective tracks, and then displays the tracks and extrapolations of the tracks on a monitor 110. Also, the computer 106 determines an appropriate action to take based on the tracks and track extensions. For example, in a commercial application at an airport, the computer can determine if two aircraft being tracked are on a collision course and if so, signal a transmitter 112 to warn each

aircraft, or if a scheduled take-off will pose the risk of collision, delay the take-off. For a military application on a ship or base, the computer can determine subsequent coordinates of enemy aircraft and send the coordinates to an  
5 antiaircraft gun or missile 120 via a communication channel 122. In a robotic application, the computer controls the robot to work on the proper object or portion of the object.

The invention generates  $k$ -dimensional matrices where  $k$  is the number of images or sets of observation data in the look  
10 ahead window plus one. Then, the invention formulates a  $k$ -dimensional assignment problem as in (0.1) above.

The  $k$ -dimensional assignment problem is subsequently relaxed to a  $(k-1)$ -dimensional problem by incorporating one set of constraints into the objective function using a  
15 Lagrangian relaxation of this set. Given a solution of the  $(k-1)$ -dimensional problem, a feasible solution of the  $k$ -dimensional problem is then reconstructed. The  $(k-1)$ -dimensional problem is solved in a similar manner, and the process is repeated until it reaches the two-dimensional  
20 problem that can be solved exactly. The ideas behind the Lagrangian relaxation scheme are outlined next.

Consider the integer programming problem

$$\begin{array}{ll}
 \text{Minimize} & v(z) = c^T z \\
 \text{Subject to:} & Az \leq b, \\
 & Bz \leq d, \\
 & z_i \text{ is an integer for } i \in I,
 \end{array} \tag{0.2}$$

where the partitioning of the constraints is natural in some sense. Given a multiplier vector  $u \geq 0$ , the Lagrangian relaxation of (0.2) relative to the constraints  $Bz \leq d$  is defined to be

$$\begin{array}{ll}
 \Phi(u) = \text{Minimize} & \{c^T z + u^T (Bz - d)\} \\
 \text{Subject to:} & Az \leq b, \\
 & z_i \text{ is an integer for } i \in I.
 \end{array} \tag{0.3}$$

If the constraint set  $Bz \leq d$  is replaced by  $Bz = d$ , the non-negativity constraint on  $u$  is removed.  $\mathcal{L} = c^T z + u^T (Bz - d)$  is the Lagrangian relative to the constraints  $Bz \leq d$ , and hence the name Lagrangian relaxation. The following fact gives the relationship between the objective functions of the original and relaxed problems.

**FACT A.1.** If  $\bar{z}$  is an optimal solution to (0.2), then  $\Phi(u) \leq v(\bar{z})$  for all  $u \geq 0$ . If an optimal solution  $\hat{z}$  of (0.3) is feasible for (0.2), then  $\hat{z}$  is an optimal solution for (0.2) and  $\Phi(u) = v(\bar{z})$ .

This leads to the following algorithm:



**Algorithm.** Construct a sequence of multipliers  $\{u_k\}_{k=0}^{\infty}$  converging to the solution  $\bar{u}$  of Maximize  $\{\Phi(u) : u \geq 0\}$  and a corresponding sequence of feasible solutions  $\{z_k\}_{k=0}^{\infty}$  of (0.2) as follows:

5        1.    Generate an initial approximation  $u_0$ .

2.    Given  $u_k$ , choose a search direction  $s_k$  and a search distance  $\alpha_k$  so that  $\Phi(u_k + \alpha_k s_k) > \Phi(u_k)$ . Update the multiplier estimate  $u_k$  by  $u_{k+1} = u_k + \alpha_k s_k$ .

3.    Given  $u_{k+1}$  and a feasible solution  $\hat{z}_{k+1}(u_{k+1})$  of (0.3),  
10    recover a feasible solution  $z_{k+1}(u_{k+1})$  of the integer programming problem (0.2).

4.    Check the termination criteria. If the algorithm is not finished, set  $k=k+1$  and return to Step 2. Otherwise, terminate the algorithm.

15        If  $\bar{z}$  is an optimal solution of (0.2), then

$$\Phi(u_k) \leq \Phi(\bar{u}) \leq v(\bar{z}) \leq v(z_k).$$

Since the optimal solution  $\hat{z} = \hat{z}(u)$  of (0.3) is usually not a feasible solution of (0.2) for any choice of the multipliers,  $\Phi(\bar{u})$  is usually strictly less than  $v(\bar{z})$ . The difference  $v(\bar{z}) -$   
20     $\Phi(\bar{u})$  is called the "duality gap," which can be estimated by comparing the best relaxed and recovered solutions computed in the course of maximizing  $\Phi(u)$ .

Thus, to eliminate the "<" in the constraint formulations (e.g., (0.1)(a) - (0.1)(d)) that resulted from false positives, the constraint sets have been modified, as indicated above, to incorporate a gap filler in each observation set to account for false positives. Thus, a zero for an index  $i_p$  ( $1 \leq p \leq k$ ) in  $z_{i_1 \dots i_k}^k$  in problem (0.4) below indicates that the track extension represented by  $z_{i_1 \dots i_k}^k$  includes a false positive in the  $p^{\text{th}}$  observation set. Note that this implies that a hypothesis be formed incorporating an observation with  $k-1$  gap fillers, e.g.,  $z_{0 \dots 0 i_p 0 \dots 0}$ ,  $i_p \neq 0$ . Thus, the resulting generalization of Problem Formulation (0.1) without the "less than" complication within the constraints is the following  $k$ -Dimensional Assignment Problems in which  $k \geq 3$ :

$$\begin{aligned}
& \text{Minimize} && \sum_{i_1=0}^{N_1} \dots \sum_{i_k=0}^{N_k} c_{i_1 \dots i_k} z_{i_1 \dots i_k} \\
& \text{Subject to:} && \sum_{i_2=0}^{N_2} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k} = 1, \quad i_1 = 1, \dots, N_1, \\
& && \sum_{i_1=0}^{N_1} \dots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k} = 1, && (0.4) \\
& && \text{for } i_j = 1, \dots, N_j \\
& && \text{and } j=2, \dots, k-1, \\
& && \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} z_{i_1 \dots i_k} = 1, \quad i_k = 1, \dots, N_k, \\
& && z_{i_1 \dots i_k} \in \{0, 1\}, \quad \forall i_n, \quad n=1, \dots, k,
\end{aligned}$$

10 where  $\mathbf{c}$  and  $\mathbf{z}$  are similarly dimensioned matrices representing costs and hypothetical assignments. Note, in general, for tracking problems these matrices are sparse.

After formulating the  $k$ -dimensional assignment problem as in (0.4), the present invention solves the resulting problem  
 15 so as to generate the outputs required by devices 110, 112, 122 and 130. For each observation set  $\mathbf{O}_i$  received from converter 104 at time  $t_i$  where  $i=1, \dots, N_1$ , the computer processes  $\mathbf{O}_i$  in a batch together with the other observation sets  $\mathbf{O}_{i-k+1}, \dots, \mathbf{O}_i$  and the track  $\mathbf{T}_{i-k}$ , i.e.,  $\mathbf{T}_j$  is the set of all  
 20 tracks that have been defined up to but not including  $\mathbf{O}_j$ . (Note, bold type designations refer to the vector of elements

for the indicated time, i.e., the set of all observations in the scan or tracks existing at the time, etc.) The result of this processing is the new set of tracks  $T_{i-k+1}$  and a set of cost weighted possible solutions indicating how the tracks might extend to the current time  $t_i$ . At time  $t_{i+1}$  the batch process is repeated using the newest observation set and deleting the oldest. Thus, there is a moving window of observation sets which is shifted forward to always incorporate the most recent observation set. The effect is that input observation sets are reused for  $(k-1)$  cycles and then on the observation set's  $k^{th}$  reuse each observation within the observation set is integrated into a track.

Figure 7 illustrates various processes implemented upon receipt of each observation set. Except for the addition of the  $k$ -dimensional assignment solving process 300 and the modification to scoring process 154 to build data structures suitable for process 300, all processes in Figure 7 are based upon prior art. The following processes 150 and 152 extend previously defined tracks  $h_{i-1}$  based on new observations. Gate formulation and output process 156 determines, for each of the previously defined tracks, a zone wherein the track may potentially extend based on limits of velocity, maneuverability and radar precision. One such technique to

accomplish this is the cone method described previously. The definition of the zone is passed to gating process 150. When a new observation set  $O_i$  is received, the gating process 150 will match each member observation with the zone for each member of the hypothetical set  $h_{i-1}$ . After all input observations from  $O_i$  are processed, the new hypothesis set  $h_i$  is generated by extending each track of the prior set of hypothetical tracks  $h_{i-1}$  either with missed detect gap fillers or with all new observation elements satisfying the track's zone. This is a many-to-many matching in that each hypothesis member can be extended to many new observations and each new observation can be used to extend many hypotheses. It, however, is not a full matching in that any hypothesis will neither be matched to all observations nor vice versa. It is this matching characteristic that leads to the sparse matrices involved in the tracking process. Subsequently, gating forwards the new hypothesis set  $h_i$  to filtering process 152. Filtering process 152 determines a smooth curve for each member of  $h_i$ . Such a smooth curve is more likely than a sharp turn from each point straight to the next point. Further, the filtering process 152 removes small errors that may occur in generating observations. Note that in performing these tasks, the filtering process 152 preferably utilizes a minimization

of a least squares test of the points in a track hypothesis or a Kalman filtering approach.

As noted above, the foregoing track extension process requires knowledge of a previous track. For the initial  
5 observations, the following gating process 158 and filtering process 160 determine the "previous track" based on initial observations. In determining the initial tracks, the points from the first observation set form the beginning points of all possible tracks. After observation data from the next  
10 observation set is received, sets of simple two-point straight line tracks are defined. Then, promotion, gate formulation, and output step 162 determines a zone in which future extensions are possible. Note that filtering step 160 uses curve fitting techniques to smooth the track extensions  
15 depending upon the number of prior observations that have accumulated in each hypothesis. Further note that promotion, gate formulation and output process 162 also determines when sufficient observations have accumulated to form a basis for promoting the track to processes 150 and 152 as described  
20 above.

The output of each of the filtering processes 152 and 160 is a set of hypothetical track extensions. Each such extension contains the hypothetical initial conditions (from

the previous track), the list of observations incorporated in the extension, and distance between each observation and the filtered track curve. Scoring process 154 determines the figure of merit or cost of an observation being an extension  
5 of a track. In one embodiment, the cost is based on the above-mentioned distance although the particular formula for determining the cost is not critical to the present invention. A preferred formula for determining the cost utilizes a negative log likelihood function in which the cost is the  
10 negative of the sum of: (a) the logs of the distances normalized by sensor standard deviation parameters, and (b) the log likelihoods for events related to track initiation, track termination, track maneuver, false negatives and false positives. Note that track maneuvers are detected by  
15 comparing the previous track curve with the current extension. Further note that some of the other events related to, for example, false negatives and false positives are detected by analyzing the relative relationship of gap fillers in the hypothesis. Thus, after determining that one of these events  
20 occurred, a cost for it can be determined based upon suitable statistics tables and system input parameters. The negative log likelihood function is desirable because it permits effective integration of the useful components. Copies of the

set of hypothetical track extensions which are scored are subsequently passed directly to one of the gate formulation and output steps 156 and 162. Note that the scoring process 154 also arranges the actual scores in a sparse matrix based upon observation identifiers, and passes them to  $k$ -dimensional assignment problem solving process 300.

The assignment solving process 300 is described below. Its output is simply the list of assignments which constitute the most likely solution of the problem described by Equation (0.4). Note that both gate formulation and output processes 156 and 162 use (at different times) the list of assignments to generate the updated track history  $T_i$  to eliminate or prune alternative previous hypotheses that are prohibited by the actual assignments in the list, and, subsequently, to output any required data. Also note that when one of the gate formulation and output processes 156 and 162 accomplish these tasks, the process will subsequently generate and forward the new set of gates for each remaining hypothesis and the processes will then be prepared to receive the next set of observations. In one embodiment, the loop described here will generate zones for a delayed set of observations rather than the subsequent set. This permits processes 156 and 162 to operate on even observation sets while the scoring step 154



and  $k$ -dimensional solving process operate on odd sets of observations, or vice versa.

The assignment solving process 300 permits the present invention to operate with a window size of dimension  $k-1$  for  
5 some  $k \geq 3$ . The upper limit on  $k$  depends only upon the computational power of the computer 106 and the response time constraints of system 100. The  $k-1$  observation sets within the processing window plus the prior track history result in a  $k$ -dimensional Assignment Problem as described by Problem  
10 Formulation (0.4). The present invention solves this generalized problem including the processes required to consider false positives and negatives, and also the processes required to consider sparse matrix problem formulations.

15 I. A FIRST EMBODIMENT OF THE  
 $k$ -DIMENSIONAL ASSIGNMENT SOLVER 300

In describing a first embodiment of the  $k$ -dimensional assignment solver 300, it is worthwhile to also discuss the process of Figure 4 which is used by the solver 300. Figure  
20 4 illustrates use of the Frieze and Yadagar process as shown in prior art for transforming a three-dimensional assignment problem into a two-dimensional assignment problem and then use a hill climbing algorithm to solve the three-dimensional assignment problem. The solver 300 uses a Lagrangian

Relaxation technique (well known in the art) to reduce the dimension of an original  $k$ -dimensional assignment problem ( $k > 3$ ) down to a three-dimensional problem and then use the process of Figure 4 to solve the three-dimensional problem.

5 Further note that the Lagrangian Relaxation technique is also utilized by the process of Figure 4 and that in using this technique the requirement that each point is assigned to one and only one track is relaxed. Instead, an additional cost, which is equal to a respective Lagrangian Coefficient  $u$ , is  
 10 added to the cost or objective function (0.1)(a) whenever a point is assigned to more than one track. This additional cost can be picked to weight the significance of each constraint violation differently, so this additional cost is represented as a vector of coefficients  $u$  which are correlated  
 15 with respective observation points. Hill climbing will then develop a sequence of Lagrangian Coefficients sets designated  $(u_0, \dots, u_j, u_{j+1}, \dots, u_p)$ . That correspond to an optimum solution of the two-dimensional assignment problem. The assignments at this optimum solution are then used to "recover" the  
 20 assignment solution of the three-dimensional assignment problem.

In step 200 of Figure 4, initial values are selected for the  $u_0$  coefficients. Because the Lagrangian Relaxation process

is iterative, the initial values are not critical and are all initially selected as zero. In step 202, these additional costs are applied to the objective function (0.1)(a). With the addition of the costs  $u$ , the goal is still to assign the

5 points which minimize the total cost. This transforms Equation (0.1)(a), written for  $k=3$  and altered to exclude mechanisms related to false positives and negatives, into objective function (1.1)(a). In the first iteration it is not necessary to consider the  $u$  matrix because all  $u$  values are

10 set to zero. To relax the requirement that each point be assigned to one and only one track, the constraint Equation (0.1)(d) is deleted, thereby permitting points from the last image to be assigned to more than one track. Note that while any axis can be chosen for relaxation, observation constraints

15 are preferably relaxed. The effect of this relaxation is to create a new problem which must have the same solution in the first two axes but which can have a different solution in the third axis. The result is constraints (1.1)(b-d).

- (a) Minimize 
$$\sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} \{(c_{i_1 i_2 i_3} - u_{i_3})\} z_{i_1 i_2 i_3}$$
- (b) Subject to: 
$$\sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} = 1, \quad i_1 = 1, \dots, N_1, \quad (1.1)$$
- (c) 
$$\sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_2 i_3} \leq 1, \quad i_2 = 1, \dots, N_2,$$
- (d) 
$$z_{i_1 i_2 i_3} \in \{0, 1\} \quad \forall \quad z_{i_1 i_2 i_3}.$$

5

Step 204 then generates from the three-dimensional problem described by Problem Formulation (0.1) a new two-dimensional problem formulation which will have the same solution for the first two indices. As Problem Formulation (1.1) has no constraints on the 3<sup>rd</sup> axis, any value within a particular 3<sup>rd</sup> axis can be used in a solution, but using anything other than the minimum value from any 3<sup>rd</sup> axis has the effect of increasing solution cost. Conceptually, the effect of step 204 is to change the three-dimensional arrays in Problem Formulation (1.1) into two-dimensional arrays as shown in Problem Formulation (1.2) and to generate the new two-dimensional matrix  $m_{i_1 i_2}$  defined as shown in Equation (1.3).

- (a) Minimize 
$$\sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} \left\{ \begin{array}{l} \text{Min:} \\ \forall i_3 \end{array} (c_{i_1 i_2 i_3} - u_{i_3}) \right\} z_{i_1 i_2}$$
- (b) Subject to: 
$$\sum_{i_2=1}^{N_2} \sum_{i_3=1}^{N_3} z_{i_2 i_3} = 1, \quad i_1 = 1, \dots, N_1, \quad (1.2)$$
- (c) 
$$\sum_{i_1=1}^{N_1} \sum_{i_3=1}^{N_3} z_{i_1 i_3} \leq 1, \quad i_2 = 1, \dots, N_2,$$
- (d) 
$$z_{i_1 i_2} \in \{0, 1\} \quad \forall z_{i_1 i_2},$$

5

$$m_{i_1 i_2} = \text{Minimum} \{ c_{i_1 i_2} - u_{j_i} \mid i=1, \dots, N_k \}. \quad (1.3)$$

The cost or objective function for the reduced problem as defined by (1.2) (a), if evaluated at all possible values of  $\mathbf{u}$  is a surface over the domain of  $\mathbf{u}$ . This surface is referred to as  $\phi(\mathbf{u})$  and is non-smooth but provable convex (i.e., it has a single peak and several other critical characteristics which form terraces). Due to the convex characteristics of  $\phi(\mathbf{u})$ , the results from solving Problem Formulation (1.2) at any particular  $\mathbf{u}_j$  can be used to generate a new set of coefficients  $\mathbf{u}_{j+1}$  whose corresponding cost value is closer to the peak of  $\phi(\mathbf{u})$ . The particular set of Lagrangian Coefficients that will generate the two-dimensional problem resulting in the maximum cost is designated  $\mathbf{u}_p$ . To recover the solution to the three-

dimensional assignment problem requires solving the Equation (1.2) problem corresponding to  $u_p$ .

In step 206, the two-dimensional problem is solved directly using a technique known to those skilled in the art such as Reverse Auction for the corresponding cost and solution values. This is the evaluation of one point on the surface or for the first iteration  $\phi(u_0)$ .

Thus, after this first iteration, the points have been assigned based on all "u" values being arbitrarily set to zero. Because the "u" values have been arbitrarily assigned, it is unlikely that these assignments are correct and it is likely that further iterations are required to properly assign the points. Step 208 determines whether the points have been properly assigned after the first iteration by determining if for this set of assignments whether a different set of "u" values could result in a higher total cost. Thus, step 208 is implemented by determining the gradient of objective function (1.2) (a) with respect to  $u_j$ . If the gradient is substantially non-zero (greater than a predetermined limit) then the assignments are not at or near enough to the peak of the surface  $\phi(u)$  (decision 210), and the new set of Lagrangian Coefficients  $u_{j+1}$  is determined.

Hill climbing Step 212 determines the  $u_{j+1}$  values based upon the  $u_j$  values, the direction resulting from projecting the previous gradient into the  $U^3$  domain, and a step size. The solution value of the two-dimensional problem is the set of

5 coefficients that minimize the two-dimensional problem and the actual cost at the minimum. Those coefficients augmented by the coefficients stored in  $m_{i_1 i_2}$  permit the evaluation (but not the minimization) of the cost term in Problem Formulation (1.1). These two cost terms are lower and upper bounds on the

10 actual minimized cost of the three-dimensional problem, and the difference between them in combination with the gradient is used to compute the step size.

With this new set of  $u$  values, steps 202-210 are repeated as a second iteration. Steps 212 and 202-210 are repeated

15 until the gradient as a function of  $u$  determined in step 208 is less than the predetermined limit. This indicates that the  $u_p$  values which locate the peak area of the  $\Phi(u)$  surface are determined and that the corresponding Problem Formulation (1.2) has been solved. Step 214 will attempt to use the

20 assignments that resulted from this particular two-dimensional assignment problem to recover the solution of the three-dimensional assignment problem as described below. If the limit was chosen properly so that the  $u$  values are close

enough to the peak, this recovery will yield the proper set of assignments that rigidly satisfies the constraint that each point be assigned to one and only one track. However, if the  $u$  values are not close enough to the peak, then the limit value for decision 210 is reduced and the repetition of steps 212 and 202-210 is continued.

Step 214 recovers the three-dimensional assignment solution by using the assignment values determined on the last iteration through step 208. Consider the two-dimensional  $z$  assignment matrix to have 1's in the locations specified by the list  $L_1 = (a_i, b_i)_{i=1}^N$ . If the three-dimensional  $z$  matrix is specified by placing 1's at the location indicated by the list  $L_2 = (a_i, b_i, m_{a_i b_i})_{i=1}^N$  then the result is a solution of Problem Formulation (1.1). Let  $L_3 = (m_{a_i b_i})_{i=1}^N$  be the list formed by the third index. If each member of  $L_3$  is unique then the  $L_2$  solution satisfies the third constraint so it is a solution to Problem Formulation (0.1). When this is not the case, recovery determines the minimal substitutions required within list  $L_3$  so that it plus  $L_1$  will be a feasible solution, i.e., a solution which satisfies the constraints of a problem formulation, but which may not optimize the objective function of the problem formulation. This stage of the recovery process is formulated as a two-dimensional assignment problem



as follows. Form a new cost matrix  $[c_{i,j}]_{j=1}^{N_i}$  where  $c_{i,j}=c_{a_i b_i, j}$  for  $j=1 \dots N_i$  and the  $N_i$  term is the total number of cost elements in the selected row of the three-dimensional cost matrix. Attempt to solve this two-dimensional problem for the minimum  
 5 using two constraints sets. If a feasible solution is found then the result will have the same form as list  $L_i$ . Replace the first set of indices by the indicated  $(a_i, b_i)$  pairs taken from list  $L_i$  and the result will be a feasible solution of Problem Formulation (0.4). If no feasible solution to the new  
 10 two-dimensional problem exists then further effort to locate the peak of  $\Phi(\mathbf{u})$  is required.

#### I.1. Generalization to a Multi-Dimensional Assignment Solving Process

15 Let  $M$  be a fixed integer and assume that  $M$  is the dimension of the initial assignment problem to be solved. Thus, initially, the result of the scoring step 154 is a  $M$ -dimensional Cost Matrix which is structured as a sparse matrix (i.e., only a small percentage of the entries in the cost and  
 20 assignment matrices are filled or non-zero). Individual cost elements represent the likelihood that a track  $\mathbf{T}_i$  as extended by the set of observations  $\{\mathbf{O}_{ij} | i = 1, \dots, M-1\}$ , is not valid. Because the matrix is sparse the list of cost elements is

stored as a packed list, and then for each dimension of the matrix, a vector of a variable length list of pointers to the cost elements is generated and stored. This organization means that for a particular observation  $O_{ij}$  for the  $j^{th}$  list in  
5 the  $i^{th}$  vector will be a list of pointers to all hypotheses in which  $O_{ij}$  participates. This structure is further explained in the following section dealing with problem partitioning.

The objective of the assignment solving process is to select from the set of all possible combinations of track  
10 extensions a subset that satisfies two criteria. First, each point in the subset of combinations should be assigned to one and only one track and therefore, included in one and only one combination of the subset, and second, the total of the scoring sums for the combinations of the subset should be  
15 minimized. This yields the following  $M$ -dimensional equations where  $k = M$ :

- (a) Minimize  $v_k(z^k) = \sum_{i_1=0}^{N_1} \dots \sum_{i_k=0}^{N_k} c_{i_1 \dots i_k}^k z_{i_1 \dots i_k}^k$  (1.4)
- (b) Subject to:  $\sum_{i_2=0}^{N_2} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^k = 1, \quad i_1 = 1, \dots, N_1,$
- (c)  $\sum_{i_1=0}^{N_1} \dots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^k = 1,$   
for  $i_j = 1, \dots, N_j$  and  $j = 2, \dots, k-1,$
- (d)  $\sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} z_{i_1 \dots i_k}^k = 1, \quad i_k = 1, \dots, N_k,$
- (e)  $z_{i_1 \dots i_k}^k \in \{0, 1\}$  for all  $i_1, \dots, i_k,$

and where  $c^k$  is the cost matrix  $[c_{i_1 \dots i_k}^k]$  which is a function of the distance between the observed point  $z^k$  and the smoothed track determined by the filtering step, and  $v_k$  is the cost function. This set of equations is similar to the set presented in Problem Formulation (0.4) except that it includes the subscript and superscript  $k$  notation. Thus, in solving the  $M$ -dimensional Assignment Problem the invention reduces this problem to an  $(M-1)$ -dimensional Assignment Problem and then to an  $(N-2)$ -dimensional Assignment Problem, etc. Further, the symbol  $k \in \{3, \dots, M\}$  customizes Problem Formulation (1.4) to a particular relaxation level. That is, the notation is used to reference data from levels relatively removed as in  $c^{k+1}$  are the cost coefficients which existed prior to this level

of relaxed coefficients  $c^k$ . Note that actual observations are numbered from 1 to  $N_i$ , where  $N_i$  is the number of observations in observation set  $i$ . Further note that the added 0 observation in each set of observations is the unconstrained  
5 "gap filler." This element serves as a filler in substituting for missed detects, and a sequence of these elements including only one true observation represents the possibility that the observation is a false positive. Also note that by being unconstrained a gap filler may be used in as many hypotheses  
10 as required.

While direct solution to (1.4) would give the precise assignment, the solution of  $k$ -dimensional equations directly for large  $k$  is too complex and time consuming for practice. Thus, the present invention solves this problem indirectly.

15 The following is a short description of many aspects of the present invention and includes some steps according to the prior art. The first step in solving the problem indirectly is to reduce the complexity of the problem by the previously known and discussed Lagrangian Relaxation technique.  
20 According to the Lagrangian Relaxation technique, the absolute requirement that each point is assigned to one and only one track is relaxed such that for some one image, points can be assigned to more than one track. However, a penalty based on

a respective Lagrangian Coefficient  $u^k$  is added to the cost function when a point in the image is assigned to more than one track. The Lagrangian Relaxation technique reduces the complexity or "dimension" of the formulation of the assignment problem because constraints on one observation set are relaxed. Thus, the Lagrangian Relaxation is performed iteratively to repeatedly reduce the dimension until a two-dimensional penalized cost function problem results as in Problem Formulation (1.1). This two-dimensional problem is solved then directly by a previously known technique such as Reverse Auction. The penalized cost function for the two-dimensional problem defines a valley or convex shaped surface which is a function of various sets of  $\{u^k | k=3, \dots, M\}$  penalty values and one set of assignments for the points in two dimensions. That is, for each particular  $u^3$  there is a corresponding two-dimensional penalized cost function problem and its solution. Note that the solution of the two-dimensional penalized cost function problem identifies the set of assignments for the particular  $u^3$  values that minimize the penalized cost function. However, these assignments are not likely to be optimum for any higher dimensional problem because they were based on an initial arbitrary set of  $u_k$  values. Therefore, the next step is to determine the optimum

assignments for the related three-dimensional penalized cost function problem. There exists a two-dimensional hill shaped function  $\Phi$  which is a graph of the minimums of all penalized cost functions at various sets of assignments in two dimensions. For the three-dimensional problem, the function  $\Phi$  can be defined based on the solution to the foregoing two-dimensional penalized cost function. By using the current  $u^3$  values and the  $\{u^k | k > 3\}$  values originally assigned, the gradient of the hill-shaped function  $\Phi$  is determined, which points toward the peak of the hill. By using the gradient and  $u^3$  values previously selected for the one point on the hill (corresponding to the minimum of the penalized cost function  $\Phi$ ) as a starting point, the  $u^3$  values can be found for which the corresponding problem will result in the peak of the function  $\Phi$ . The solution of the corresponding two-dimensional problem is the proper values for two of the three sets of indices in the three-dimensional problem. These solution indices can select a subsection of the cost array which maps to a two-dimensional array. The set of indices which minimize the two-dimensional assignment problem based on that array corresponds to the proper assignment of points in the third dimension. The foregoing "recovery" process was known in the prior art, but it is modified here to adjust for the sparse

matrix characteristic. The next task is to recover the solution of the proper  $u^4$  values for the four-dimensional problem. The foregoing hill climbing process will not work again because the foregoing hill climbing process when

5 required to locate the four-dimensional  $u^4$  values for the peak of  $\phi^3$  requires the exact definition of the function  $\phi^3$  (as was available in the case of  $\phi^2$ ) or an always less than approximation of  $\phi^3$ , whereas the iteration can result in a greater than approximation of  $\phi^3$ . According to the present

10 invention, another three-dimensional function  $\psi$  is defined which is a "less than approximation" the three-dimensional function  $\phi$  and which can be defined based on the solution to the two-dimensional penalized cost function and the previously assigned and determined  $u^k$  values. Next, the gradient of the

15 function  $\psi$  is found and hill climbing technique used to determine the  $u^4$  values at the peak. Each selected  $u^4$  set results in a new three-dimensional problem and requires the two-dimensional hill climbing based upon new two-dimensional problems. At the peak of the three-dimensional function  $\psi$ ,

20 the solution values are a subset of the values required for the four-dimensional solution. Recovery processing extends the subset to a complete solution. This process is repeated iteratively until the  $u^k$  values that result in a corresponding

solution at the peak of the highest order functions  $\Psi$  and  $\Phi$  are found. The final recovery process then results in the solution of  $k$ -dimensional problem.

Figure 5 illustrates process 300 in more detail.

5

## I.2. Problem Formulation

In problem formulation step 310, all data structures for the subsequent steps are allocated and linked by pointers as required for execution efficiency. The incoming problem is  
 10 partitioned as described in the subsequent section. This partitioning has the effect of dividing the incoming problem into a set of independent problems and thus reducing the total workload. The partitioning process depends only on the actual cost matrix so the partitioning can and is performed for all  
 15 levels of the relaxation process.

### I.2.1. Relaxation and Recovery

Step 320 begins the Lagrangian Relaxation process for reducing the  $M$ -dimensional problem by selecting all Lagrangian  
 20 Coefficient  $u^M$  penalty values initially equal to zero. The Lagrangian Coefficients associated with the  $M^{th}$  constraint set are an  $(N_M+1)$ -element vector. The reduction of this  $M$ -dimensional problem in step 322 to a  $(M-1)$ -dimensional problem



uses the two step process described above. First, a penalty based on the value of the respective  $u^M$  coefficient is added to the cost function when a point is assigned to more than one track and then the resultant cost function is minimized.

5 However, during the first iteration, the penalty is zero because all  $u^M$  values are initially set to zero. Second, the requirement that no point from any image can be assigned to more than one track is relaxed for one of the images. In the extreme this would allow a point from the relaxed image to be  
10 associate with every track. However, the effect of the previous penalty would probably mean that such an association would not minimize the cost. The effect of the two steps in combination is to remove a hard constraint while adding the penalty to the cost function so that it operates like a soft  
15 constraint. For step 322 this two-step process results in the following penalized cost function problem with  $k=M$ :

(a)  $\Phi_k(u^k) \equiv \text{Minimize: } \phi_k(u^k, z^k)$

$$= \sum_{i_1=0}^{N_1} \dots \sum_{i_k=0}^{N_k} c_{i_1 \dots i_k}^k z_{i_1 \dots i_k}^k - \sum_{i_k=0}^{N_k} u_{i_k}^k \left[ \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} z_{i_1 \dots i_{k-1} i_k}^k - 1 \right]$$

$$= \sum_{i_1=0}^{N_1} \dots \sum_{i_k=0}^{N_k} (c_{i_1 \dots i_k}^k - u_{i_k}^k) z_{i_1 \dots i_k}^k + \sum_{i_k=0}^{N_k} u_{i_k}^k$$

(b) Subject to:  $\sum_{i_2=0}^{N_2} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^k = 1, \quad i_1 = 1, \dots, N_1, \quad (1.5)$

5 (c)  $\sum_{i_1=0}^{N_1} \dots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^k = 1$   
for  $i_j = 1, \dots, N_j$   
and  $j=2, \dots, k-1,$

(d)  $z_{i_1 \dots i_k}^k \in \{0, 1\}$  for all  $i_1 \dots i_k.$

10 Because the constraint on single assignment of elements from  
the last image has been eliminated, an  $(M-1)$ -dimensional  
problem can be developed by eliminating some of the possible  
assignments. As shown in Equations (1.6), this is done by  
selecting the smallest cost element from each of the  $M^{th}$  axis  
15 vectors of the cost matrix. Reduction in this manner yields  
a new, lower-order penalized cost function defined by  
Equations (1.6) which has the same minimum cost as does the  
objective function defined by (1.5) (a) above.

The cost vectors are selected as follows. Define a new  
20 cost array  $c_{i_1 \dots i_{k-1}}^{k-1}$  by:

$$\begin{aligned}
c_{i_1 \dots i_k}^k &= \text{Minimum} \left\{ c_{i_1 \dots i_k}^k - u_{i_k}^k \mid i_k = 0, 1, \dots, N_k \right\}, \\
&\text{for } (i_1, \dots, i_{k-1}) \neq (0, \dots, 0), \\
c_{0 \dots 0}^{k-1} &= \sum_{i_k=0}^{N_k} \min \left\{ 0, c_{0 \dots 0}^k - u_{i_k}^k \right\}.
\end{aligned} \tag{1.6}$$

The resulting  $(M-1)$ -dimensional problem is (where  $k=M$ ):

$$\begin{aligned}
\Phi_k(u^k) &= \text{Minimize: } v_{k-1}(z^{k-1}) = \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} c_{i_1 \dots i_{k-1}}^{k-1} z_{i_1 \dots i_{k-1}}^{k-1} \\
\text{Subject to: } &\sum_{i_2=0}^{N_2} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^{k-1} = 1, \quad i_1 = 1, \dots, N_1, \\
&\sum_{i_1=0}^{N_1} \dots \sum_{i_{j-1}=0}^{N_{j-1}} \sum_{i_{j+1}=0}^{N_{j+1}} \dots \sum_{i_k=0}^{N_k} z_{i_1 \dots i_k}^{k-1} = 1, \\
&\quad \text{for } i_j = 1, \dots, N_j \\
&\quad \text{and } j=2, \dots, k-1, \\
&\sum_{i_1=0}^{N_1} \dots \sum_{i_{k-2}=0}^{N_{k-2}} z_{i_1 \dots i_{k-1}}^{k-1} = 1, \quad i_{k-1} = 1, \dots, N_{k-1}, \\
&z_{i_1 \dots i_k}^k \in \{0, 1\} \quad \text{for all } i_1 \dots i_{k-1}.
\end{aligned} \tag{1.7}$$

- 10 Assignment Problem (1.4) and Problem Formulation (1.7) differ  
only in the dimension  $M$  vs.  $M-1$ , respectively. An optimal  
solution to (1.7) is also an optimal solution to equation  
(1.5). This relationship is the basis for an iterative  
sequence of reductions indicated by steps 320-332 through 330-  
15 332 and 200-204 in which the penalized cost function problem  
is reduced to a two-dimensional problem. As these formula  
will be used to describe the processing at all levels, the

lowercase  $k$  is used except where specific reference to the top level is needed. In step 206, the two-dimensional penalized cost function is solved directly by the prior art Reverse Auction technique. Each execution of 206 produces two  
 5 results, the set of  $\mathbf{z}^2$  values that minimize the problem and the cost that results  $v^2$  when these  $\mathbf{z}$  values are substituted into the objective function (1.7)(a).

In step 208, according to the prior art, solution  $\mathbf{z}^2$  values are substituted into the two-dimensional derivative of  
 10 the surface  $\phi_2$ . The result indicates how the value of  $\mathbf{u}^3$  should be adjusted so as to perform the hill climbing function. As was previously described the objective is to produce a sequence of  $\mathbf{u}_i^3$  values which ends when the  $\mathbf{u}_p^3$  value is in the domain of the peak of the surface  $\phi$ . The section  
 15 "Determining Effective Gradient" describes how new values are computed and how it is determined that the current  $\mathbf{u}_i^k$  points to the peak of  $\phi_2$ . When no further adjustment is required the flow moves to step 214 which will attempt to recover the three-dimensional solution as previously described. When  
 20 further adjustment is required then the flow progresses to step 212 and the new values of  $\mathbf{u}^k$  are computed. At the two-dimensional level the method of the prior art could be used for the hill climbing procedure. However, it is not practical

to use this prior art hill climbing technique to determine the updated Lagrangian Coefficients  $u^k$  or the Max on the next (or any) higher order surface  $\Phi$  because the next (or any) higher dimensional function  $\Phi$  cannot be defined based on known  
 5 information.

Instead, the present invention defines a new function based on known information which is useful for hill climbing from the third to the fourth and above dimensions, i.e.,  $u^k$  values which result in  $z$  values that are closer to the proper  
 10  $z$  values for the highest  $k$ -dimension. This hill climbing process (which is different than that used in the prior art of Figure 4 for recovering only the three-dimensional solution) is used iteratively at all lower dimensions  $k$  of the  $M$ -dimensional problem (including the three-dimensional level  
 15 where it replaces prior art) even when  $k$  is much larger than three. Figure 6 helps to explain this new hill climbing technique and illustrates the original  $k$ -dimensional cost function  $v_k$  of Problem Formulation (1.4). However, the actual  $k$ -dimensional cost surface  $v_k$  defined by (1.4)(a) comprises  
 20 scalar values at each point described by  $k$ -dimensional vectors and as such can not be drawn. Nevertheless, for illustration purposes only, Figure 6 ignores the reality of ordering vectors and illustrates a concave function  $v_k(z^k)$  to represent

Equation (1.4). The surface  $v_k$  is illustrated as being smooth to simplify the explanation although actually it can be imagined to be terraced. The goal of the assignment problem is to find the values of  $\bar{z}^k$ ; these values minimize the  $k$ -dimensional cost function  $v_k$ .

For purposes of explanation, assume that in Figure 6,  $k=4$  (the procedure is used for all  $k \geq 3$ ). This problem is reduced by two iterations of Lagrangian Relaxation to a two-dimensional penalized cost function  $\phi_j^{(k-1)}{}_i$ . This cost function, and all other cost functions described below, are also non-smooth and continuous but are illustrated in Figure 6 as smooth for explanation purposes. Solving the  $\phi_j^{(k-1)}{}_i$  problem results in one set of  $z^2$  assignments and the value of  $\phi_{(k-1)}{}_i$  at the point  $u^2_{ij}$ . A series of functions  $\phi_j^{(k-1)}{}_i, \dots, \phi_1^{(k-1)}{}_i$  each generated from a different  $u^3$  are shown. The particular series illustrates the process of locating the peak of  $\phi_{(k-1)}{}_i$ . The two-dimensional penalized cost functions  $\phi_j^{(k-1)}{}_i, \dots, \phi_1^{(k-1)}{}_i$  can be solved directly. Each such solution provides the information required to calculate the next  $u^3$  value. Each iteration of the hill climbing improves the selection of  $u^3$  values, i.e., yields  $\phi^2$  problem whose solution is closer to those at the solution of the  $\phi^3$  problem. The result of solving  $\phi_1^{(k-1)}{}_i$  is values that are on both  $\phi_{(k-1)}{}_i$  and  $\phi_k$ . Figure 6

illustrates the surface  $\phi_k$  which comprises the minimums of all  $k$ -dimensional penalized cost function surfaces  $\phi_k$ , i.e., if the Problem Formulations (1.5) and (1.7) were solved at all possible values of  $u^k$  the function  $\phi_k(u_k)$  would result. The

5 surface  $\phi_k$  is always less than the surface  $v_k$  except at the peak as described in Equation (1.8) and its maximum occurs where the surface  $v_k$  is minimum. Because the surface  $\phi_k$  represents the minimum of the surface  $\phi_k$ , any point on the surface  $\phi_k$  can be mapped to the surface  $v_k$ . The function  $\phi_k$

10 provides a lower bound on the minimization problem described by (1.4)(a). Let  $\bar{z}^k$  be the unknown solution to Problem Formulation (1.4) and note that:

$$\phi^k(u^k) \leq v_k(\bar{z}^k) \leq v_{k-1}(z^k). \quad (1.8)$$

15 Consequently, the values  $z^k$  at the peak of  $\phi_k$  (i.e., the greatest lower bound on the cost of the relaxed problem), can be substituted into the  $k$ -dimensional penalized cost function to determine the proper assignments. Consequently, the present invention attempts to find the maximum on the surface

20  $\phi_k$ . However, it is not possible to use the prior art hill climbing to climb to the peak of  $\phi_k$  because the definition of  $\phi_k$  requires exact knowledge of lower order functions  $\phi$ . As the solution of  $\phi_1^k$  is not the exact solution, in that higher

order values of  $\mathbf{u}$  are not yet optimized, its value can be larger than the true peak of  $\phi_k$ . As such it is not a lower bound on  $v_k$  and it can not be used to recover the required solution.

5        Instead, the present invention defines all auxiliary functions  $\psi_k$  which are based only on the solution to the two-dimensional penalized cost function problem, lower order values  $\mathbf{z}^k$  and values  $\mathbf{u}^k$  determined previously by the reduction process. The function  $\psi_k$  is a less than approximation of  $\phi_k$ ,  
 10        and its gradient is used for hill climbing to its peak. The values  $\mathbf{z}^k$  at the peak of the function  $\psi_k$  are then substituted into Problem Formulation (1.5) to determine the proper assignments. To define the function  $\psi_k$ , the present invention explicitly makes the function  $\phi_k(\mathbf{u}^k)$  a function of all higher  
 15        order sets of Lagrangian Coefficients with the expanded notation:  $\phi_k(\mathbf{u}^k; \mathbf{u}^{k+1}, \dots, \mathbf{u}^K)$ . Then, a new set of functions  $\psi$ , is defined recursively, using the  $\phi_k$ 's domain:

$$\psi_3(u^3) = v_2 + \sum_{i_3=0}^{N_3} u_{i_3}^3 = \phi_3(u^3; u^4, \dots, u^K), \quad (1.9)$$

20

where  $v_2$  is the solution value for the most recent two-dimensional penalized cost function problem. For  $k > 3$



$$\Psi_k(u^3, \dots, u^{k-1}; u^k) = \begin{cases} \Phi_k(u^k; u^{k+1}, \dots, u^K), & \text{if known,} \\ \Psi_{k-1}(u^3, \dots, u^{k-2}; u^{k-1}) + \sum_{i_k=0}^{N_k} u_{i_k}^k, & \text{otherwise.} \end{cases} \quad (1.10)$$

From the definitions of  $\Phi_k$  and  $v_k$  (Problem Formulation (1.7) compared with Problem Formulation (1.5)):

$$\Phi_k(u^k; u^{k+1}, \dots, u^M) = v_{k-1}(z^{k-1}) + \sum_{i_k=0}^{N_k} u_{i_k}^k,$$

it follows that:

$$\Psi_3(u^3) = v_2 + \sum_{i_3=0}^{N_3} u_{i_3}^3 = \Phi_3(u^3; u^4, \dots, u^M) \leq v_3^3(z^3)$$

and with that Equation (1.8) is extended to:

$$\Psi_k(u^3, \dots, u^{k-1}; u^k) \leq \Phi_k(u^k; u^{k+1}, \dots, u^M) \leq v_k(\bar{u}^k) \leq v_k(u^k) \quad (1.11)$$

- 5 This relationship means that either  $\Phi_k$  or  $\Psi_k$  may be used in hill climbing to update the Lagrangian Coefficients  $u$ .  $\Phi_k$  is the preferred choice, however it is only available when the solution to Problem Formulation (1.5) is a feasible solution to Problem Formulation (1.4) (as in hill climbing from the

second to third dimension which is why prior art worked). For simplicity in implementation, the function  $\psi_k$  is defined so that it equals  $\phi_k$  when either function could be used. It is therefore always used, even for hill climbing from the second  
 5 dimension. The use of the function  $\psi$  which is based on previously assigned or determined  $u$  values and not higher order  $u$  values which are not yet determined, is an important feature of the present invention.

#### 10 I.2.2. Determining Effective Gradient

After the function  $\psi_k$  is defined, the next steps of hill climbing/peak detection are to determine the gradient of the function  $\psi_k$ , determine an increasing portion of the gradient and then move up the surface  $\psi_k$  in the direction of this  
 15 increasing portion of the gradient. As shown in Figure 6, any upward step on the surface  $\psi_k$ , for example, to the minimum of the  $\phi^k_j$  will yield a new set of values  $u^k$  (to the "left") that is closer to the ideal set of values  $u^k$  which correspond to the minimum of the function  $v_k$ . While it is possible to  
 20 iteratively step up this surface  $\psi_k$  with steps of fixed size and then determine if the peak has been reached, the present invention optimizes this process by determining the single step size from the starting point at the minimum of  $\phi^k_i$  that

will jump to the peak and then calculate the values  $u^k$  at the peak. Once the values  $u$  at the peak of  $\psi_k$  are determined, then the values  $u^k$  can be substituted into Problem Formulation (1.5) to determine the proper assignment. (However, in a more realistic example, where  $k$  is much greater than three, then the values  $u^k$  at the peak of the function  $\psi$  along with the lower-order values  $u^k$  and those assigned and yielded by the reduction steps are used to define the next higher level function  $\psi$ . This definition of a higher order function  $\psi$  and hill climbing process are repeated iteratively until  $\psi_k$ , the peak of  $\psi_k$ , and the values  $u^k$  at the peak of  $\psi_k$  are identified.) The following is a description of how to determine the gradient of each surface  $\psi$  and how to determine the single step size to jump to the peak from the starting point on each surface  $\psi$ .

As noted above, each surface  $\psi$  is non-smooth. Therefore, if a gradient was taken at a single point, the gradient may not point toward the peak. Therefore, several gradients (a "bundle") are determined at several points on the surface  $\psi_k$  in the region of the starting point (i.e., minimum of  $\phi_{k_i}$ ) and then averaged. Statistically, the averaged result should point toward the peak of the surface  $\psi_k$ . Wolfe's Conjugate Subgradient Algorithm (P. Wolfe. A method of conjugate

subgradients for minimizing non-differentiable functions. *Mathematical Programming Study*, 3:145-173, 1975; P. Wolfe. Finding the nearest point in a polytope. *Mathematical Programming Study*, 11:128-149, 1976) for minimization was  
 5 previously known in another environment to determine a gradient of a non-smooth surface using multiple subgradients and can be used with modification in the present invention. The modification to Wolfe's algorithm uses the information generated for  $\psi_k(\mathbf{u}^{k3}, \dots, \mathbf{u}^{kk-2}; \mathbf{u}^{kk-1})$  as the basis for calculating  
 10 the subgradients. The definition of a subgradient  $\mathbf{v}$  of  $\psi_k(\mathbf{u}^{k3}, \dots, \mathbf{u}^{kk})$  is any member of the subdifferential set defined as:

$$\partial\psi_k(\mathbf{u}) = \{ \mathbf{v} \in \mathbb{R}^{N_k+1} \mid (\psi_k(\mathbf{u}^{k3}, \dots, \mathbf{u}^{kk-1}; \mathbf{u}') - \psi_k(\mathbf{u}^{k3}, \dots, \mathbf{u}^{k-1}; \mathbf{u}^k)) \geq \mathbf{v}^T(\mathbf{u}' - \mathbf{u}^k) \quad \forall \mathbf{u}' \in \mathbb{R}^{N_k+1} \},$$

where  $\mathbf{v}^T$  is the transpose of  $\mathbf{v}$ .

Next, a subgradient vector is determined from this function. If  $\mathbf{z}^k$  is the solution of Problem Formulation (1.5), then differentiating  $\psi_k(\mathbf{u}^3, \dots, \mathbf{u}^{k-1}; \mathbf{u}^k)$  with respect to  $\mathbf{u}_{ik}^k$  and  
 20 evaluating the result with respect to the current selection matrix  $\mathbf{z}^k$  yields a subgradient vector:

$$g = (0, (1 - \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} z_{i_1 \dots i_k}^k | i_k = 1, \dots, N_k)) .$$

The iterative nature of the solution process at each dimension yields a set of such subgradients. Except for a situation described below, where the resultant averaged gradient does not point toward the peak, the most recent set of such subgradients are saved and used as the "bundle" for the peak finding process for this dimension. For example, at the level  $k$  there is a bundle of subgradients of the surface  $\psi_k$  near the minimum of the surface  $\phi_i^k$  determined as a result of solving Problem Formulation (1.4) at all lower levels. This bundle can be averaged to approximate the gradient. Alternately, the previous bundle can be discarded so as to use the new value to initiate a new bundle. This choice provides a way to adjust the process to differing classes of problems, i.e., when data is being derived from two sensors and the relaxation proceeds from data derived from one sensor to the other then the prior relaxation data for the first sensor could be detrimental to performance on the second sensors data.

### I.2.3. Step Size and Termination Criterion

After the average gradient of the surface  $\psi_k$  is determined, the next step is to determine a single step that will jump to the peak of the surface  $\psi_k$ . The basic strategy is first to specify an arbitrary step size, and then calculate the value of  $\psi_k$  at this step size in the direction of the gradient. If the value of  $\psi_k$  is larger than the previous one, this probably means that the step has not yet reached the peak. Consequently, the step size is doubled and a new value of  $\psi_k$  is determined and compared to the previous one. This process is repeated until the new value of  $\psi_k$  is less than the previous one. At that time, the step has gone too far and crossed over the peak. Consequently, the last doubling is rolled-back and that step size is used for the iteration. If the initial estimated  $\psi_k$  is less than previous value then the step size is decreased by 50%. If this still results in a smaller value of  $\psi_k$ , then the last step is rolled back and the previous step size is decreased by 25%. The following is a more detailed description of this process.

With a suitable bundle of subgradients determined as just described, Wolfe's algorithm can be used to determine the effective subgradient  $d$  and the upgraded value  $u_{j+1}^k$ . From the

previous iteration, or from an initial condition, there exists a step length value  $t$ . The value,

$$\mathbf{u}_+ = \mathbf{u}_j^k + t\mathbf{d}$$

is calculated as an estimate of  $\mathbf{u}_{j+1}^k$ . To determine if the  
 5 current step size is valid we evaluate  $\psi_k(\mathbf{u}^3, \dots, \mathbf{u}^{k-2}; \mathbf{u}_+)$ . If  
 the result represents an improvement then we double the step  
 size. Otherwise we halve the step size. In either case a new  
 $\mathbf{u}_+$  is calculated. The doubling or halving continues until the  
 step becomes too large to improve the result, or until it  
 10 becomes small enough to not degrade the result. The resulting  
 suitable step size is saved with  $\mathbf{d}$  as part of the subgradient  
 bundle. The last acceptable  $\mathbf{u}_+$  is assigned to  $\mathbf{u}_{j+1}^k$ .

Three distinct criteria are used to determine when  $\mathbf{u}_j^k$  is  
 close enough to  $\bar{\mathbf{u}}^k$ :

- 15 1. The Wolfe's algorithm criterion of  $\mathbf{d}=0$  given that the  
 test has been repeated with the bundle containing only  
 the most recent subgradient.
2. The difference between the lower bound  $\phi_k(\mathbf{u}^k)$  and the  
 upper bound  $v_k(\mathbf{z}^k, \mathbf{u}^k)$  being less than a preset relative  
 20 threshold. (Six percent was found to be an effective  
 threshold for radar tracking problems.)
3. An iteration count being exceeded.

The use of limits on the iteration are particularly effective for iterations at the level 3 through (n-1) as these iterations will be repeated so as to resolve higher order coefficient sets. With limited iterations the process is in  
 5 general robust enough to improve the estimate of upper order Lagrangian Coefficients. By limiting the iteration counts then the total processing time for the algorithm becomes deterministic. That characteristic means the process can be effective for real time problems such as radar, where the  
 10 results of the last scan of the environment must be processed prior to the next scan being received.

#### I.2.4. Solution Recovery

The following process determines if the  $u^k$  values at what  
 15 is believed to be the peak of the function  $\psi_k$  adequately approximate the  $u^k$  values at the peak of the corresponding function  $\phi_k$ . This is done by determining if the corresponding penalized cost function is minimized. Thus, the  $u^k$  values at the peak of the function  $\psi_k$  are first substituted into Problem  
 20 Formulation (1.5) to determine a set of  $z$  assignments for  $k-1$  points. During the foregoing reduction process, Problem Formulation (1.7) yielded a tentative list of  $k$  selected  $z$  points that can be described by their indices as:



$\left\{ \left( i_1^j \dots i_{k-1}^j \right) \right\}_{j=1}^{N_0}$ , where  $N_0$  is the number of cost elements selected into the solution. One possible solution of Problem Formulation (1.4) is the solution of Problem Formulation (1.5) which is described as  $\left\{ \left( i_1^j \dots i_{k-1}^j m_{i_1 \dots i_{k-1}}^k \right) \right\}_{j=1}^{N_0}$  with  $m_{i_1 \dots i_{k-1}}^k$  as it was defined in Problem Formulation (1.6). If this solution satisfies the  $k^{th}$  constraint set, then it is the optimal solution.

However, if the solution for Problem Formulation (1.5) is not feasible (decision 355), then the following adjustment process determines if a solution exists which satisfies the  $k^{th}$  constraint while retaining the assignments made in solving Problem Formulation (1.7). To do this, a two-dimensional cost matrix is defined based upon all observations from the  $k^{th}$  set which could be used to extend the relaxed solution.

$$h_{jl} = c_{i_1, \dots, i_{k-1}, l}^k \quad \begin{array}{l} \text{for } l=0, \dots, N_k \\ \text{and } j=0, \dots, N_0. \end{array} \quad (1.12)$$

If the resulting two-dimensional assignment problem,

$$\begin{aligned}
&\text{Minimize} && \sum_{j=0}^{N_0} \sum_{l=0}^{N_l} h_{jl} w_{jl} \\
&\text{Subject to:} && \sum_{l=0}^{N_k} w_{jl} = 1 \quad j=1, \dots, N_0, \\
&&& \sum_{l=0}^{N_k} w_{jl} = 1 \quad j=1, \dots, N_0, \\
&&& w_{jl} \in \{0, 1\} \quad j=0, \dots, N_0, \quad l=0, \dots, N_k
\end{aligned} \tag{1.13}$$

5

has a feasible solution, then the indices of that solution map to the solution of Problem Formulation (1.4) for the  $k$ -dimensional problem. The first index in each resultant solution entry is the pointer back to an element of the

10  $\left\{ \left( i_1^j \dots i_{k-1}^j m_{i_1 \dots i_{k-1}} \right) \right\}_{j=1}^{N_0}$  list. That element supplies the first  $k-1$  indices of the solution. The second index of the solution to the recovery problem is the  $k^{\text{th}}$  index of the solution. Together these indexes specify the values of  $z^k$  that solve Problem Formulation (1.4) at the  $k^{\text{th}}$  level.

15

If Problem Formulation (1.13) does not have a feasible solution, then the value of  $u_i^k$  which was thought to represent  $u_p^k$  is not representative of the actual peak and further iteration at the  $k^{\text{th}}$  level is required. This decision represent the branch path from step 214 and equivalent steps.

20

### I.3. Partitioning

A partitioning process is used to divide the cost matrix that results from the Scoring step 154 into as many independent problems as possible prior to beginning the relaxation solution. The partitioning process is included with the Problem Formulation Step 310. The result of partitioning is a set of problems to be solved, i.e., there will be  $p_1$  problems that consist of a single hypothesis,  $p_2$  problems that consist of two hypothesis, etc. Each such problem is a group in that one or more observations or tracks are shared between members of the group.

In partitioning to groups no consideration is given to the actual cost values. The analysis depends strictly on the basis of two or more cost elements sharing the same specific axis of the cost matrix. In a two-dimensional case two cost elements must be in the same group if they share a row or a column. If the two elements are in the same row, then each other cost element that is also in the same row, as well as any cost elements that are in columns occupied by members of the row must be included in the group. The process continues recursively. In literature it is referred to as "Constructing a Spanning Forest." The  $k$ -dimensional case is analogous to the two-dimensional case. The specific method we have

incorporated is a depth first search, presented by Aho, Hopcroft, and Ullman (A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Design and Analysis of Computer Algorithms*. Addison-Wesley, MA, 1974).

5       The result of partitioning at level  $M$  is the set of problems described as  $\{P_{ij} | i=1, \dots, p_j \text{ and } j=1, \dots, N\}$ , where  $N$  is the total number of hypothesis. The problems labeled  $\{P_{i1} | i=1, \dots, p_1\}$  are the cases where there is only one choice for the next observation at each scan and that observation  
10 could be used for no other track, i.e., it is a single isolated track. The hypothesis must be included in the solution set and no further processing is required.

As hypothesis are constructed the first element is used to refer to the track ID. Any problem in the partitioned set  
15 which does not have shared costs in the first scan represent a case where a track could be extended in several ways but none of the ways share an observation with any other track. The required solution hypothesis for this case is the particular hypothesis with the maximum likelihood. For this  
20 case all likelihoods are determined as was described in Scoring and the maximum is selected.

In addition to partitioning at the level  $M$ , partitioning is applied to each subsequent level  $M-1, \dots, 2$ . For each

problem that was not eliminated by the prior selection, partitioning is repeated, ignoring observations that are shared in the last set of observations. Partitioning recognizes that relaxation will eliminate the last constraint  
5 set and thus partitioning is feasible for the lower level problems that will result from relaxation. This process is repeated for all levels down to  $k=3$ . The full set of partitionings can be performed in the Problem Formulation Step 310, prior to initiating the actual relaxation steps. The  
10 actual two-dimensional solver used in step 206 includes an equivalent process so no advantage would be gained by partitioning at the  $k=2$  level.

There are two possible solution methods for the remaining problems. "Branch and Bound" as was previously described, or  
15 the relaxation method that this invention describes. If any of the partitions have 5-10 possible tracks and less than 50 to 20 hypotheses, then the prior art "Branch and Bound" algorithm generally executes faster than does the relaxation due to its reduced level of startup overhead. The "Branch and  
20 Bound" algorithm is executed against all remaining  $M$  level problems that satisfy the size constraint. For the remainder the Relaxation algorithm is used. The scheduling done in Problem Formulation allows each Problem Formulation (1.5) cost

matrix resulting from the first step of relaxation to be partitioned. The resulting partitions can be solved by any of the four methods: isolated track direct inclusion, isolated track tree evaluation, small group "Branch and Bound" or an  
 5 additional stage of relaxation as has been fully described.

The partitioning after each level of Lagrangian Relaxation is effective because when a problem is relaxed, the constraint that requires each point to be assigned to only one track is eliminated (for one image at a time). Therefore, two  
 10 tracks previously linked by contending for one point will be unlinked by the relaxation which permits the point to be assigned to both tracks. The fruitfulness of partitioning increases for lower and lower levels of relaxation.

The following is a more detailed description of the  
 15 partitioning method. Its application at all but the level  $M$  depends upon the relaxation process described in this invention. The recursive partitioning is therefore a distinct part of this invention. The advantage of this method is greatly enhanced by the sparse cost matrix resulting from  
 20 tracking problems. However the sparse nature of the problem requires special storage and search techniques.

A hypothesis is the combination of the cost element  $c_n$  the selection variable  $z_n$  and all observations that made up the

potential track extension. It can be written as,  
 $h_n = \{c_n, z_n, \{o_{nk} = o_{k_i} | k=1, \dots, M_i \in \{1, \dots, N_k\}\}\}$ , i.e., cost,  
 selection variable and observations in the hypothesis. Here  
 $n \in \{1, \dots, N\}$ , where  $N$  is the total number of hypotheses in the  
 5 problem. While the cost and assignment matrices were  
 previously referenced, these matrices are not effective  
 storage mechanisms for tracking applications. Instead the  
 list of all hypothesis and sets of lists for each dimension  
 that reference the hypothesis set are stored. The  
 10 hypothetical set in list form is:

$$\{h_n\}_{n=1}^{n=N}.$$

For each dimension  $k=1, \dots, M$  there exists a set of lists,  
 with each list element being a pointer to a particular  
 15 hypothesis:

$$L_{k_i} = \{p_{k_j} | k_j=1, \dots, N_{k_i}\}_{i=1, k=1}^{i=N_{k_i}, k=M},$$

where  $N_{k_i}$  is a number of hypothesis containing the  $i^{th}$   
 observation from scan  $k$ . This structure is a multiply linked  
 list in that any observation is associated with a set of  
 pointer to all hypothesis it participates in, and any  
 20 hypothesis has a set of pointers to all observations that

formed it. (These pointers can be implemented as true pointers or indices depending upon the particular programming language utilized.)

Given this description of the matrix storage technique  
 5 then the partitioning technique is as follows: Mark the first  
 list  $L_{k_i}$  and follow out all pointers in that list to the  
 indicated hypothesis  $h_{p_{k_i}}$  for  $i=1, \dots, N_{k_i}$ . Mark all located  
 hypothesis, and for each follow pointers back the particular  
 $L_k$  for  $k=1, \dots, M$ . Those  $L$ 's if not previously marked get  
 10 marked and also followed out to hypothesis elements and back  
 to  $L$ 's. When all such  $L$ 's or  $h$ 's being located are marked,  
 then an isolated sub-problem has been identified. All marked  
 elements can be removed from the lists and stored as a unique  
 problem to be solved. The partitioning problem then continues  
 15 by again starting at the first residual set  $L$ . When none  
 remain, the original problem is completely partitioned.

Isolated problems can result from one source track having  
 multiple possible extensions or from a set of source tracks  
 contending for some observations. Because one of the indices  
 20 of  $k$  (in our implementation it is  $k=1$ ) indicates the source  
 track then it is possible to categorize the two problem types  
 by observing if the isolated problem includes more than one  
 $L$ -list from the index level associated with tracks.



#### I.4. Subsequent Track Prediction

As noted above, after the points are assigned to the respective tracks, some action is taken such as controlling take-offs and landings to avoid collision, advising airborne aircraft to change course, warning airborne aircraft of an adjacent aircraft in a commercial environment, or aiming and firing an anti-aircraft (or anti-missile) missile, rocket or projectile, or taking evasive action in a military environment. Also, the tracking can be used to position a robot to work on an object. For some or all of these applications, usually the tracks which have just been identified are extended or extrapolated to predict a subsequent position of the aircraft or other object. The extrapolation can be done in a variety of prior art ways; for example, straight line extensions of the most likely track extension hypothesis, parametric quadratic extension of the x versus time, y versus time, etc. functions that are the result of the filtering process described earlier, least square path fitting or Kalman filtering of just the selected hypothesis.

The process of gating, filtering, and gate generation as they were previously described require that a curve be fit through observations such that fit of observations to the likely path can be scored and further so that the hypothetical

target future location can be calculated for the next stage of gating. In the implementation existing, quadratic functions have been fit through the measurements that occur within the window. A set of quadratics is used, one per sensor  
5 measurement. To calculate intercept locations these quadratics can be converted directly into path functions. Intercept times are then calculated by prior methods based upon simultaneous solution of path equations.

The use of the fitted quadratics is not as precise as  
10 more conventional filters like the Kalman Filter. They are however much faster and sufficiently accurate for the relative scores required for the Assignment problem. When better location prediction is required then the assignment problem is executed to select the solution hypothesis and based upon the  
15 observations in those hypothesis the more extensive Kalman filter is executed. The result is tremendous computation savings when compared with the Kalman Filter being run on all hypothetical tracks.

Based on the foregoing, apparatus and methods have been  
20 disclosed for tracking objects. However, numerous modifications and substitutions can be made without deviating from the scope of the present invention. For example, the foregoing functions  $\psi$  can also be defined as recursive

approximation problem in which several values of higher order  $u^k$  values are used to eliminate the higher than approximation characteristic of the function  $\phi_k$ . The hill climbing of the function  $\psi$  can be implemented by a high order hill climbing using the enhanced function  $\phi_k$ . Although the result would not be as efficient it seems likely that the method would converge. Therefore, the invention has been disclosed by way of example and not limitation, and reference should be made to the following claims to determine the scope of the present invention.

## II. AN ALTERNATIVE EMBODIMENT OF THE MULTI-DIMENSIONAL ASSIGNMENT SOLVING PROCESS

The following description provides an alternative second embodiment of the multi-dimensional assignment solving process of section I.1. However, in order to clearly describe the present alternative embodiment, further discussion is first presented regarding the data assignment problem of partitioning measurements into tracks and false alarms and, in particular, the representation of multi-dimensional assignment problems.

II.1. Formulation of the Assignment Problem

The goal of this section is to explain the formulation of the data association problems, and more particularly multi-dimensional assignment problems, that govern large classes of problems in centralized or hybrid centralized-sensor level multisensor/multitarget tracking. The presentation is brief; technical details are presented for both track initiation and maintenance in (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) for non-maneuvering targets and in (A.B. Poore. Multidimensional assignments and multitarget tracking: Partitioning data sets. In P. Hansen, I.J. Cox, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 19, pages 169-198, Providence, R.I., 1995. American Mathematical Society) for maneuvering targets. Note that the present formulation can also be modified to include target features (e.g., size and type) into the scoring process 154.

The data assignment problems for multisensor and multitarget tracking considered here are generally posed as

that of maximizing the posterior probability of the surveillance region (given the data) according to

$$\text{Maximize } \left\{ \frac{P(\Gamma=\gamma|Z^M)}{P(\Gamma=\gamma^0|Z^M)} \mid \gamma \in \Gamma^* \right\}, \quad (2.1)$$

5

where  $Z^M$  represents  $M$  data sets,  $\gamma$  is a partition of indices of the data (and thus induces a partition of the data),  $\Gamma^*$  is the finite collection of all such partitions,  $\Gamma$  is a discrete random element defined on  $\Gamma^*$ ,  $\gamma^0$  is a reference partition, and  
 10  $P(\Gamma=\gamma|Z^M)$  is the posterior probability of a partition  $\gamma$  being true given the data  $Z^M$ . The term partition is defined below.

Consider  $M$  observation sets  $Z(k)$  ( $k=1, \dots, M$ ) each of  $N_k$  observations  $\{z_{i_k}^k\}_{i_k=1}^{N_k}$ , and let  $Z^M$  denote the cumulative data set defined by

$$Z(k) = \{z_{i_k}^k\}_{i_k=1}^{N_k} \text{ and } Z^M = \{Z(1), \dots, Z(M)\}, \quad (2.2)$$

15 respectively. In multisensor data fusion and multitarget tracking the data sets  $Z(k)$  may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are measurements that must be partitioned into tracks and false alarms. In the formulation  
 20 of track extensions a moving window over time of observations

sets is used. The observation sets will be measurements which are: (a) assigned to existing tracks, (b) designated as false measurements, or (c) used for initiating tracks. However, note that alternative data objects instead of observation sets  
 5 may also be fused such as in sensor level tracking wherein each sensor forms tracks from its own measurements and then the tracks from the sensors are fused in a central location. Note that, as one skilled in the art will appreciate, both embodiments of the present invention may be used for this type  
 10 of data fusion.

The data assignment problem considered presently is represented as a case of set partitioning defined in the following way. First, for notational convenience in representing tracks, a zero index is added to each of the  
 15 index sets in (2.2), a gap filler  $z_0^k$  is added to each of the data sets  $Z(k)$  in (2.2), and a "track of data" is defined as  $\left( z_{i_1}^1, \dots, z_{i_N}^N \right)$ , where  $i_k$  can now assume the values of 0. A partition of the data will refer to a collection of tracks of data or track extensions wherein each observation occurs  
 20 exactly once in one of the tracks of data and such that all data is used up. Note that the occurrence of the gap filler is unrestricted. The gap filler  $z_0^k$  serves several purposes in the representation of missing data, false observations,

initiating tracks, and terminating tracks. Note that a "reference partition" may be defined which is a partition in which all observations are declared to be false.

Next under appropriate independence assumptions it can be  
5 shown that:

$$\frac{P(\Gamma=\gamma|Z^M)}{P(\Gamma=\gamma^0|Z^M)} \equiv L_\gamma \equiv \prod_{(i_1, \dots, i_M) \in \gamma} L_{i_1, \dots, i_M}, \quad (2.3)$$

where  $L_{i_1, \dots, i_M}^k$  is a likelihood ratio containing probabilities for  
10 detection, maneuvers, and termination as well as probability density functions for measurement errors, track initiation and termination. Then if  $c_{i_1, \dots, i_M}^k = -\ln(L_{i_1, \dots, i_M}^k)$ ,

$$-\ln \left[ \frac{P(\Gamma=\gamma|Z^M)}{P(\Gamma=\gamma^0|Z^M)} \right] = \sum_{(i_1, \dots, i_M) \in \gamma} c_{i_1, \dots, i_M}^k. \quad (2.4)$$

15

Using (2.3) and the zero-one variable  $z_{i_1, \dots, i_M}^k = 1$ , if  $(i_1, \dots, i_M) \in \gamma$ , and 0, otherwise, one can then write the problem (2.1) as the following  $M$ -dimensional assignment problem (as also presented in section Problem Formulation

20 (1.4) with  $k=M$ ):

- (a) Minimize 
$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_M=0}^{N_M} c_{i_1 \dots i_M} z_{i_1 \dots i_M}$$
- (b) Subject to: 
$$\sum_{i_2=0}^{N_2} \cdots \sum_{i_M=0}^{N_M} z_{i_1 \dots i_M} = 1, \quad i_1 = 1, \dots, N_1,$$
- (c) 
$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \cdots \sum_{i_M=0}^{N_M} z_{i_1 \dots i_M} = 1, \quad (2.5)$$
  
for  $i_k = 1, \dots, N_k$  and  $k=2, \dots, M-1,$
- 5 (d) 
$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{M-1}=0}^{N_{M-1}} z_{i_1 \dots i_M} = 1, \quad i_M = 1, \dots, N_M,$$
- (e) 
$$z_{i_1 \dots i_M} \in \{0, 1\} \quad \text{for all } i_1, \dots, i_M,$$

where  $c_0 \dots_0$  is arbitrarily defined to be zero. Here, each group of sums in the constraints (2.5) (b) - (2.5) (e) represents  
 10 the fact that each non-gap filler observation occurs exactly once in a "track of data." One can modify this formulation to include multi-assignments of one, some, or all the actual observations. The assignment problem (2.5) is changed accordingly. For example, if  $z_{i_k}^k$  is to be assigned no more  
 15 than, exactly, or no less than  $n_{i_k}^k$  times, then the " $=1$ " is changed to " $\leq, =, \geq n_{i_k}^k$ ," respectively.

Expressions for the likelihood ratios  $L_{i_1 \dots i_M}^k$  appearing in the costs  $c_{i_1 \dots i_M} = -\ln(L_{i_1 \dots i_M})$  are well-known. In particular, discussions of these ratios may be found in (A.B. Poore.  
 20 Multidimensional assignment formulation of data association



problems arising from multitarget tracking and multisensor data fusion, *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking: Partitioning data sets. In P. Hansen, I.J. Cox, B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 19, pages 169-198, Providence, R.I., 1995. American Mathematical Society). Furthermore, the likelihood ratios are easily modified to include target features and to account for different sensor types. Also note that in track initiation, the  $M$  observation sets provide observations from  $M$  sensors, possibly all the same. Additionally note that for track maintenance, a sliding window of  $M$  observation sets and one data set containing established tracks may be used. In this latter case, the formulation is the same as above except that the dimension of the assignment problem is now  $M+1$ .

## II.2. Overview of the New Lagrangian Relaxation Algorithms

Having formulated an  $M$ -dimensional assignment problem (2.5), we now turn to a description of the Lagrangian relaxation algorithm. Subsection II.2.1 below presents many of the relaxation properties associated with the relaxation of an  $n$ -dimensional assignment problem to an  $m$ -dimensional one

via a Lagrangian relaxation of  $n-m$  sets of constraints wherein  
 $m < n \leq M$  and preferably in the present invention embodiment  
 $n-m > 1$ . Although any  $n-m$  constraint sets can be relaxed, the  
description here is based on relaxing the last  $n-m$  sets of  
5 constraints and keeping the first  $m$  sets. Given either an  
optimal or suboptimal solution of the relaxed problem, a  
technique for recovering a feasible solution of the  
 $n$ -dimensional problem is presented in subsection II.2.2 below  
and an overview of the Lagrangian relaxation algorithm is  
10 given in subsection II.2.3.

The following notation will be used throughout the  
remainder of the work. Let  $M$  be an integer such that  $M \geq 3$  and  
let  $n \in \{3, \dots, M\}$ . The  $n$ -dimensional assignment problem is

$$\begin{aligned}
 & \text{(a) Minimize: } v_n(z) = \sum_{i_1=0}^{N_1} \dots \sum_{i_n=0}^{N_n} c_{i_1 \dots i_n}^n z_{i_1 \dots i_n}^n \\
 & \text{(b) Subject To } \sum_{i_2=0}^{N_2} \dots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n = 1, \quad i_1 = 1, \dots, N_1, \quad (2.6) \\
 & \text{(c) } \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \dots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n = 1 \\
 & \quad \text{for } i_k = 1, \dots, N_k \text{ and } k = 2, \dots, n-1, \\
 & \text{(d) } \sum_{i_1=0}^{N_1} \dots \sum_{i_{n-1}=0}^{N_{n-1}} z_{i_1 \dots i_n}^n = 1, \quad i_n = 1, \dots, N_n, \\
 & \text{(e) } z_{i_1 \dots i_n}^n \in \{0, 1\} \text{ for all } i_1, \dots, i_n.
 \end{aligned}$$

To ensure that a feasible solution of (2.6) always exists, all variables with exactly one nonzero index (i.e., variables of the form  $z_0^n \dots o_{i_k 0} \dots_0$  for  $i_k \neq 0$ ) are assumed free to be assigned and the corresponding cost coefficients are well-defined.

5

### II.2.1. The Lagrangian Relaxed Assignment Problem

The  $n$ -dimensional assignment problem (2.6) has  $n$  sets of constraints. A  $(N_k+1)$ -dimensional multiplier vector associated with the  $k^{th}$  constraint set will be denoted by  $u^k = (u_0^k, u_1^k, \dots, u_{M_k}^k)^T$  with  $u_0^k = 0$  and  $k=1, \dots, n$ . The  $n$ -dimensional assignment problem (2.6) is relaxed to an  $m$ -dimensional assignment problem by incorporating  $n-m$  of the  $n$  sets of constraints into the objective function (2.6) (a). Although any constraint set can be relaxed, the description of the relaxation procedure for (2.6) will be based on the relaxation of the last  $n-m$  sets of constraints. The relaxed problem is

$$\Phi_{mn}(u^{m+1}, \dots, u^n) \equiv \text{Minimize } \phi_{mn}(z^n; u^{m+1}, \dots, u^n) \quad (2.7)$$

$$\begin{aligned} &\equiv \sum_{i_1=0}^{N_1} \dots \sum_{i_n=0}^{N_n} c_{i_1 \dots i_n} z_{i_1 \dots i_n}^n \\ &+ \sum_{k=m+1}^n \sum_{i_k=0}^{N_k} u_{i_k}^k \left[ \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \dots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n - 1 \right] \end{aligned}$$

$$\equiv \sum_{i_1=0}^{N_1} \cdots \sum_{i_n=0}^{N_n} \left[ c_{i_1 \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k \right] z_{i_1 \dots i_n}^n - \sum_{k=m+1}^n \sum_{i_k=0}^{N_k} u_{i_k}^k$$

Subject To 
$$\sum_{i_2=0}^{N_2} \cdots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n = 1, \quad i_1 = 1, \dots, N_1,$$

$$\sum_{i_1=0}^{N_1} \cdots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \cdots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n = 1$$

for  $i_k = 1, \dots, N_k$  and  $k = 2, \dots, m,$

$$z_{i_1 \dots i_n}^n \in \{0, 1\} \quad \text{for all } i_1, \dots, i_n,$$

wherein we have added the multiplier  $u_0^k \equiv 0$  for notational convenience. Thus, the multiplier  $u^k \in \mathbb{R}^{N_k+1}$  with  $u_0^k = 0$  is fixed.

An optimal (or suboptimal) solution of (2.7) can be  
 5 constructed from that of an  $m$ -dimensional assignment problem.  
 To show this, define for each  $(i_1, \dots, i_m)$  an index  $(j_{m+1}, \dots, j_n)$   
 $= (j_{m+1}(i_1, \dots, i_m), \dots, j_n(i_1, \dots, i_m))$  and a new cost function  
 $c_{i_1 \dots i_m}^m$  by

$$(j_{m+1}, \dots, j_n) =$$

$$= \operatorname{argmin} \left\{ c_{i_1 \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k \mid i_k = 0, \dots, N_k \text{ and } k = m+1, \dots, n \right\}, \quad (2.8)$$

$$c_{i_1 \dots i_m}^m = c_{i_1 \dots i_m j_{m+1} \dots j_n}^n + \sum_{k=m+1}^n u_{j_k}^k \quad \text{for } (i_1, \dots, i_m) \neq (0, \dots, 0),$$

$$c_{0\dots 0}^m = \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_n=0}^{N_n} \text{Minimum} \left\{ 0, c_{0\dots 0i_{m+1}\dots i_n}^n + \sum_{k=m+1}^n u_{j_k}^k \right\}.$$

If  $(j_{m+1}, \dots, j_n)$  is not unique, choose the smallest such  $j_{m+1}$ , amongst those  $(n-m)$ -tuples with the same  $j_{m+1}$  choose the smallest  $j_{m+2}$ , etc., so that  $(j_{m+1}, \dots, j_n)$  is uniquely defined.)

Using the cost coefficients defined in this way, the following

5  $m$ -dimensional assignment problem is obtained:

$$\Phi_{mn}(u^{m+1}, \dots, u^n) = \text{Minimize } \hat{\Phi}_{mn}(z^m; u^{m+1}, \dots, u^n) \equiv v_m(z^m)$$

$$\equiv \sum_{i_1=0}^{N_1} \dots \sum_{i_m=0}^{N_m} c_{i_1\dots i_m}^m z_{i_1\dots i_m}^m$$

$$\text{Subject to: } \sum_{i_2=0}^{N_2} \dots \sum_{i_m=0}^{N_m} z_{i_1\dots i_m}^m = 1, \quad i_1 = 1, \dots, N_1, \quad (2.9)$$

$$\sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \dots \sum_{i_m=0}^{N_m} z_{i_1\dots i_m}^m = 1,$$

for  $i_k = 1, \dots, N_k$  and  $k=2, \dots, m-1$ ,

$$\sum_{i_1=0}^{N_1} \dots \sum_{i_{m-1}=0}^{N_{m-1}} z_{i_1\dots i_m}^m = 1, \quad i_m = 1, \dots, N_m,$$

$$z_{i_1\dots i_m}^m \in \{0, 1\} \quad \text{for all } i_1, \dots, i_m.$$

As an aside, observe that any feasible solution  $z^n$  of (2.6)

15 yields a feasible solution  $z^m$  of (2.9) via the construction

$$z_{i_1\dots i_m}^m = \begin{cases} 1, & \text{if } z_{i_1\dots i_{n-1}i_n}^n = 1 \text{ for some } (i_{m+1}, \dots, i_n), \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the  $m$ -dimensional assignment problem (2.9) has at least as many feasible solutions of the constraints as the original problem (2.6).

The following Fact A.2 has been shown to be true. It states that an optimal solution of Problem Formulation (2.7) can be computed from that of Problem Formulation (2.9). Furthermore, the converse to this fact is provided in Fact A.3. Moreover, if the solution of either of these two problems (2.7) or (2.9) is  $\epsilon$ -optimal (i.e., the objective function associated with the suboptimal solution is within " $\epsilon$ " of the objective function associated with the optimal solution), then so is the other.

**Fact A.2.** Let  $w^m$  be a feasible solution to problem (2.9) and define  $w^n$  by

$$\begin{aligned}
 15 \quad w_{i_1 \dots i_n}^n &= w_{i_1 \dots i_m}^m \quad \text{if } (i_{m+1}, \dots, i_n) = (j_{m+1}, \dots, j_n) \text{ and} \\
 &\quad (i_1, \dots, i_m) \neq (0, \dots, 0) \\
 w_{i_1 \dots i_n}^n &= 0 \quad \text{if } (i_{m+1}, \dots, i_n) \neq (j_{m+1}, \dots, j_n) \text{ and} \quad (2.10) \\
 &\quad (i_1, \dots, i_m) \neq (0, \dots, 0) \\
 20 \quad w_{0 \dots 0 i_{m+1} \dots i_n}^n &= 1 \quad \text{if } c_{0 \dots 0 i_{m+1} \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k \leq 0 \\
 w_{0 \dots 0 i_{m+1} \dots i_n}^n &= 0 \quad \text{if } c_{0 \dots 0 i_{m+1} \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k > 0.
 \end{aligned}$$

Then  $w^n$  is a feasible solution of the Lagrangian relaxed problem (2.7) and

$$\phi_{mn}(w^n; u^{m+1}, \dots, u^n) = \hat{\phi}_{mn}(w^m; u^{m+1}, \dots, u^n) - \sum_{k=m+1}^n \sum_{i_k=0}^{M_k} u_{i_k}^k.$$

If, in addition,  $w^m$  is optimal for (2.9), then  $w^n$  is an optimal solution of (2.7) and

$$\phi_{mn}(u^{m+1}, \dots, u^n) = \hat{\phi}_{mn}(u^{m+1}, \dots, u^n) - \sum_{k=m+1}^n \sum_{i_k=0}^{M_k} u_{i_k}^k.$$

5 With the exception of one equality being converted to an inequality, the following Fact is a converse of Fact A.2 and has also been shown to be true.

**Fact A.3.** Let  $w^n$  be a feasible solution to problem (2.7) and define  $w^m$  by

$$\begin{aligned} 10 \quad w_{i_1 \dots i_m}^m &= \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_n=0}^{N_n} w_{i_1 \dots i_n}^n \text{ for } (i_1, \dots, i_m) \neq (0, \dots, 0), \\ w_{0 \dots 0}^m &= 0 \text{ if } (i_1, \dots, i_m) = (0, \dots, 0) \text{ and} \\ c_{0 \dots 0 i_{m+1} \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k &> 0 \text{ for all } (i_{m+1}, \dots, i_n), \\ w_{0 \dots 0}^m &= 1 \text{ if } (i_1, \dots, i_m) = (0, \dots, 0) \text{ and} \\ c_{0 \dots 0 i_{m+1} \dots i_n}^n + \sum_{k=m+1}^n u_{i_k}^k &\leq 0 \text{ for some } (i_{m+1}, \dots, i_n). \end{aligned} \quad (2.11)$$

15

Then  $w^m$  is a feasible solution of the problem (2.9) and

$$\phi_{mn}(w^n; u^{m+1}, \dots, u^n) \geq \hat{\phi}_{mn}(w^m; u^{m+1}, \dots, u^n) - \sum_{k=m+1}^n \sum_{i_k=0}^{M_k} u_{i_k}^k.$$

If, in addition,  $w^n$  is optimal for (2.7), then  $w^m$  is an optimal solution of (2.9),

$$\begin{aligned} \phi_{mn}(w^n; u^{m+1}, \dots, u^n) &= \hat{\phi}_{mn}(w^m; u^{m+1}, \dots, u^n) - \sum_{k=m+1}^n \sum_{i_k=0}^{M_k} u_{i_k}^k, \text{ and} \\ \phi_{mn}(u^{m+1}, \dots, u^n) &= \hat{\phi}_{mn}(u^{m+1}, \dots, u^n) - \sum_{k=m+1}^n \sum_{i_k=0}^{M_k} u_{i_k}^k. \end{aligned}$$

5

#### II.2.2. The Recovery Procedure

The next objective is to explain a recovery procedure or method for recovering a solution to the  $n$ -dimensional problem of (2.6) from a relaxed problem having potentially substantially fewer dimensions than (2.6). Note that this aspect of the alternative embodiment of the present invention is substantially different from the method disclosed in the first embodiment of the multidimensional assignment solving process of section I.1 of this specification. Further, this alternative embodiment provides substantial benefits in terms of computational efficiency and accuracy over the first embodiment, as will be discussed. Thus, given a feasible



(optimal or suboptimal) solution  $w^m$  of (2.9) (or  $w^n$  if Problem Formulation (2.7) is constructed via Fact A.2), an explanation is provided here regarding how to generate a feasible solution  $z^n$  of (2.6) which is close to  $w^m$  in a sense to be specified.

5 We first assume that no variables in (2.6) are preassigned to zero (this assumption will be removed shortly). The difficulty with the solution  $w^n$  is that it need not satisfy the last  $n-m$  sets of constraints in (2.6). Note, however, that if  $w^m$  is an optimal solution for (2.9) and  $w^n$  (constructed as in

10 Fact A.2) satisfies the relaxed constraints, then  $w^n$  is optimal for (2.6). The recovery procedure described here is designed to preserve the 0-1 character of the solution  $w^m$  of (2.9) as far as possible. That is, if  $w_{i_1 \dots i_m}^m = 1$  and  $i_l \neq 0$  for at least one  $l=1, \dots, m$ , then the corresponding feasible solution  $z^n$  of (2.6)

15 is constructed so that  $z_{i_1 \dots i_n}^n = 1$  for some  $(i_{m+1}, \dots, i_n)$ . Note that by this reasoning, variables of the form  $z_{0 \dots 0 i_{m+1} \dots i_n}^n$  can be assigned to a value of 1 in the recovery problem only if  $w_{0 \dots 0}^m = 1$ . However, variables  $z_{0 \dots 0 i_{m+1} \dots i_n}^n$  will be treated differently in the recovery procedure in that they can be

20 assigned 0 or 1 independent of the value  $w_{0 \dots 0}^m$ . This increases the feasible set of the recovery problem, leading to a potentially better solution.

Let  $\{(i_1^j, \dots, i_m^j)\}_{j=1}^{N_0}$  be an enumeration of indices of  $w^m$  (or the first  $m$  indices of  $w^n$  constructed as in Fact A.2) such that  $w_{i_1^j \dots i_m^j}^m = 1$  and  $(i_1^j, \dots, i_m^j) \neq (0, \dots, 0)$ . Set  $(i_1^0, \dots, i_m^0) = (0, \dots, 0)$  for  $j=0$  and define

$$5 \quad c_{j i_{m+1} \dots i_n}^{n-m+1} = c_{i_1^j \dots i_m^j i_{m+1} \dots i_n}^n \quad (2.12)$$

for  $i_k=0, \dots, N_k; k=m+1, \dots, n; j=0, \dots, N_0$ .

Let  $Y$  denote the solution of the  $(n-m+1)$ -dimensional  
10 assignment problem:

$$\begin{aligned} & \text{Minimize} \quad \sum_{j=0}^{N_0} \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_n=0}^{N_n} c_{j i_{m+1} \dots i_n}^{n-m+1} Y_{j i_{m+1} \dots i_n} \\ & \text{Subject to} \quad \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_n=0}^{N_n} Y_{j i_{m+1} \dots i_n} = 1, \quad j=1, \dots, N_0, \quad (2.13) \\ & \quad \sum_{j=0}^{N_0} \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \dots \sum_{i_n=0}^{N_n} Y_{j i_{m+1} \dots i_n} = 1, \\ & \quad \text{for } i_k=1, \dots, N_k \text{ and } k=m+1, \dots, n-1, \\ & \quad \sum_{j=0}^{N_0} \sum_{i_{m+1}=0}^{N_{m+1}} \dots \sum_{i_{n-1}=0}^{N_{n-1}} Y_{j i_{m+1} \dots i_n} = 1, \quad i_n=1, \dots, N_n, \\ & \quad Y_{j i_{m+1} \dots i_n} \in \{0, 1\} \quad \text{for all } j, i_{m+1}, \dots, i_n. \end{aligned}$$

The recovered feasible solution  $z^n$  of (2.6) corresponding to  
20 the multiplier set  $\{u^m, \dots, u^n\}$  is then defined by

$$z_{i_1 \dots i_n}^n = \begin{cases} 1, & \text{if } (i_1, \dots, i_m) = (i_1^j, \dots, i_m^j) \text{ for some} \\ & j=0, \dots, N_0 \text{ and } Y_{j i_{m+1} \dots i_n} = 1, \\ 0, & \text{otherwise.} \end{cases} \quad (2.14)$$

This recovery procedure is valid as long as all cost coefficients  $c^n$  are defined and all zero-one variables in  $z^n$  are free to be assigned. Note that modifications are necessary for sparse problems. If the cost coefficient

5  $c_{i_1^j \dots i_m^j i_{m+1} \dots i_n}^n$  is well defined and the zero-one variable  $c_{i_1^j \dots i_m^j i_{m+1} \dots i_n}^n$  is free to be assigned to zero or one, then define  $c_{j i_{m+1} \dots i_n}^{n-m+1} = c_{i_1^j \dots i_m^j i_{m+1} \dots i_n}^n$  as in (2.12) with  $z_{j i_{m+1} \dots i_n}^{n-m+1}$  being free to be assigned. Otherwise,  $z_{j i_{m+1} \dots i_n}^{n-m+1}$  is preassigned to zero. To

ensure that a feasible solution exists, we now only need  
10 ensure that the variables  $z_{j 0 \dots 0}^{n-m+1}$  are free for  $j=0, 1, \dots, N_0$ .

(Recall that all variables of the form  $z_{0 \dots i_k \dots 0}^n$  are free (to be assigned) and all coefficients of the form  $c_{0 \dots i_k \dots 0}^n$  are well defined for  $k=1, \dots, n$ .) This is accomplished as follows. If

the cost coefficient  $c_{i_1^j i_2^j \dots i_m^j 0 \dots 0}^n$  is well defined and  
15  $z_{i_1^j i_2^j \dots i_m^j 0 \dots 0}^n$  is free, then define  $c_{j 0 \dots 0}^{n-m+1} = c_{i_1^j i_2^j \dots i_m^j 0 \dots 0}^n$  with  $z_{j 0 \dots 0}^{n-m+1}$  being free. Otherwise, since all variables of the form

$z_{0 \dots i_k \dots 0}^n$  are known feasible and have well-defined costs, put

$$c_{j 0 \dots 0}^{n-m+1} = \sum_{k=1, i_k^j \neq 0}^m c_{0 \dots 0 i_k^j 0 \dots 0}^n.$$

### II.3. The Multi-Dimensional Lagrangian Relaxation Algorithm for the n-Dimensional Assignment Problem

Starting with the  $M$ -dimensional assignment problem (2.6),  
 5 i.e.,  $n=M$ , the algorithm described below is recursive in that  
 the  $M$ -dimensional assignment problem is relaxed to an  $m$ -  
 dimensional problem by incorporating  $(n-m)$  sets of constraints  
 into the objective function using Lagrangian relaxation of  
 this set. This problem is maximized with respect to the  
 10 Lagrange multipliers, and a good suboptimal solution to the  
 original problem is recovered using an  $(n-m+1)$ -dimensional  
 assignment problem. Each of these two (the  $m$ -dimensional and  
 the  $(n-m+1)$ -dimensional assignment problems) can be solved in  
 a similar manner.

15 More precisely, reference is made to Figure 8 which  
 presents a flowchart of a procedure embodying the multi-  
 dimensional relaxation algorithm, referred to immediately  
 above. This procedure, denoted MULTI\_DIM\_RELAX in Figure 8,  
 has three formal parameters,  $n$ , PROB\_FORMULATION and  
 20 ASSIGNMT\_SOLUTION, which are used to transfer information  
 between recursive instantiations of the procedure. In  
 particular, the parameter,  $n$ , is an input parameter supplying  
 the dimension for the multi-dimensional assignment problem (as  
 in(2.6)) which is to be solved (i.e., to obtain an optimal or

near-optimal solution). The parameter, PROB\_FORMULATION, is also an input parameter supplying the data structure(s) used to represent the  $n$ -dimensional assignment problem to be solved. Note that PROB\_FORMULATION at least provides access  
 5 to the cost matrix,  $[c^n]$ , and the observation sets whose observations are to be assigned. Additionally, the parameter, ASSIGNMT\_SOLUTION, is used as an output parameter for returning a solution of a lower dimensioned assignment problem to an instantiation of MULTI\_DIM\_RELAX which is processing a  
 10 higher dimensioned assignment problem.

A description of Figure 8 follows. Assuming MULTI\_DIM\_RELAX is initially invoked with  $M$  as the value for the parameter  $n$  and the PROB\_FORMULATION having a data structure(s) representing an  $M$ -dimensional assignment problem  
 15 as in (2.5), in step 500 an integer  $m$ ,  $2 \leq m < n$  is chosen such that the constraint sets corresponding to dimensions  $m+1, \dots, n$  are to be relaxed so that an  $m$ -dimensional problem formulation as in (2.7) results. In step 504 an initial approximation is provided for  $\{u_0^{m+1}, \dots, u_0^n\}$ . Subsequently, in step 508 the above  
 20 initial values for  $\{u_0^{m+1}, \dots, u_0^n\}$  are used in an iterative process in determining  $(\bar{u}^{m+1}, \dots, \bar{u}^n)$  which maximizes  $\{\Phi_{mn}(u^{m+1}, \dots, u^n)$  where  $u^k \in \mathbb{R}^{M_{k+1}}$  and  $k=m+1, \dots, n\}$  for a feasible solution  $w^n$  subject to the constraints of Problem Formulation

(2.7). Note that by maximizing  $\Phi_{mn}(u^{m+1}, \dots, u^n)$ , some of the constraints that were relaxed are being forced to be satisfied and in so doing information is built into a solution of this function for solving the input assignment problem in

5 "PROB\_FORMULATION." Further note that a non-smooth optimization technique is used here and that a preferred method of determining a maximum in step 508 is the bundle-trust method of Schramm and Zowe (H. Schramm and J. Zowe. A version of the bundle idea for minimizing a non-smooth

10 function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2, No. 1:121-152, February, 1992). This method, along with various other methods for determining the maximum in step 508, are discussed below.

15 Also note that for  $m > 2$ , a solution to the optimization problem of step 508 is NP-hard and therefore cannot be solved optimally. That is, there is no known computationally tractable method for guaranteeing an optimal solution. Thus, there are two possibilities: either (a) allow  $m$  to be greater

20 than 2 and use auxiliary functions similar to those disclosed in the first embodiment of the  $k$ -dimensional assignment solver 300 in section I to compute a near-optimal solution, or (b) make  $m=2$  wherein algorithms such as the forward/reverse

auction algorithm of D.P. Bertsekas and D.A. Castañon (D.P. Bertsekas and D.A. Castañon. A forward/reverse auction algorithm for asymmetric assignment problems. *Computational Optimization and Applications*, 1: 277-298, 1992) provides an  
 5 optimal solution.

If option (a) immediately above is chosen, then the auxiliary functions are used as approximation functions for obtaining at least a near-optimal solution to the optimization problem of step 508. Note that the auxiliary functions used  
 10 depend on the value of  $m$ . Thus, auxiliary functions used when  $m=3$  will likely be different from those for  $m=4$ . But in any case, the optimization procedure is guided by using the merit function,  $\Phi_{2n}(u^3, \dots, u^n)$ , which can be computed exactly via a two-dimensional assignment problem for guiding the  
 15 maximization process.

Alternatively, if option (b) above is chosen, then two important advantages result, namely, the optimization problem of step 508 can be always solved optimally and without using auxiliary approximation functions. Thus, better solutions to  
 20 the original  $M$ -dimensional assignment problem are likely since there is no guarantee that when the non-smooth optimization techniques are applied to the auxiliary functions the techniques will yield an optimal solution to step 508.

Furthermore, it is important to note that without auxiliary functions, the processing in step 508 is both conceptually easier to follow and more efficient.

Subsequently, in step 512 of Fig. 8, the solution  
 5  $(\bar{u}^{m+1}, \dots, \bar{u}^n)$  is used in determining an optimal solution  $w^m$  to Problem Formulation (2.9) as generated according to (2.8) and Fact A.3.

In step 516, the solution  $w^n$  is used in defining the cost matrix  $c^{n-m+1}$  as in (2.12). Subsequently, if  $n-m+1=2$ , then the  
 10 assignment problem (2.13) may be solved straightforwardly using known techniques such as forward/reverse auction algorithms. Following this, in step 528, the solution to the two-dimensional assignment problem is assigned to the variable ASSIGNMT\_SOLUTION and in step 532 ASSIGNMT\_SOLUTION is  
 15 returned to a dimension three level recursion of MULTI\_DIM\_RELAX for solving a three-dimensional assignment problem.

Alternatively, if in step 520,  $n-m+1>2$ , then in step 536 the data structure(s) representing a problem formulation as in  
 20 (2.13) is generated and assigned to the parameter, PROB\_FORMULATION. Subsequently, in step 540 a recursive copy of MULTI\_DIM\_RELAX is invoked to solve the lower dimensional assignment problem represented by PROB\_FORMULATION. Upon the



completion of step 540, the parameter, ASSIGNMT\_SOLUTION, contains the solution to the  $(n-m+1)$ -dimensional assignment problem. Thus, in step 544, the  $(n-m+1)^{\text{th}}$  solution is used to solve the  $n$ -dimensional assignment problem as discussed  
 5 regarding equations (2.14). Finally, in steps 548 and 552 the solution to the  $n$ -dimensional assignment problem is returned to the calling program so that, for example, it may be used in taking one or more actions such as (a) sending a warning to aircraft or sea facility; (b) controlling air traffic; (c)  
 10 controlling anti-aircraft or anti-missile equipment; (d) taking evasive action; or (e) surveilling an object.

### II.3.1. Comments on the Various Procedures Provided by Figure 8

15 There are many procedures described by Figure 8. One such procedure is the first embodiment of the multidimensional assignment solving process of section I.1. That is, by specifying  $m=n-1$  in step 500, a single set of constraints is  
 20 relaxed in step 508. Thus, one set of constraints is incorporated into the objective function via the Lagrangian problem formulation, resulting in an  $(m=n-1)$ -dimensional problem. The relaxed problem is subsequently maximized in step 512 with respect to the corresponding Lagrange  
 25 multipliers and then a feasible solution is reconstructed for

the  $n$ -dimensional problem using a two-dimensional assignment problem. The second procedure provided by Figure 8 is a novel approach which is not suggested by the first embodiment of section I.1. In fact, the second procedure is somewhat of a mirror image of the first embodiment in that  $n-2$  sets of constraints are simultaneously relaxed, yielding immediately an  $(m=2)$ -dimensional problem in step 512. Thus, a feasible solution to the  $n$ -dimensional problem is then recovered using a recursively obtained solution to an  $(n-1)$ -dimensional problem via step 540. In this case, the function values and subgradients of  $\Phi_{2n}(u^1, \dots, u^n)$  of step 508 can be computed optimally via a two-dimensional assignment problem. The significant advantage here is that there is no need for the merit or auxiliary functions,  $\Psi_k$ , as required in the first embodiment of the multidimensional assignment solving process of section I.1 above and also there is no need for the more general merit or auxiliary functions  $\Psi_m$ , as discussed in subsection II.4.2 below. Further, note that all function values and subgradients used in the nonsmooth maximization process are computed exactly (i.e., optimally) in this second procedure. Moreover, problem decomposition is now carried out for the  $n$ -dimensional problem; however, decomposition of the  $(n-1)$ -dimensional recovery problem (and all lower recovery

problems) is performed only after the problem is formulated.

Between these two procedures are a host of different relaxation schemes based on relaxing  $n - m$  sets of constraints to an  $m$ -dimensional problem ( $2 < m < n$ ), but these all have the same difficulties as the procedure for the first embodiment of section I.1 in that the relaxed problem is an NP-hard problem. To resolve this difficulty, we use an auxiliary or merit function  $\Psi_{mn}$  as described in subsection II.4.2 below. (The notation  $\Psi_k$  was used for  $\Psi_{n-1,n}$  in section I.1.) For the case  $m < n-1$ , the recovery procedure is still based on an NP-hard  $(n-m+1)$ -dimensional assignment problem. Note that the partitioning techniques similar to those discussed in Section I.3 may be used to identify the assignment problem with a layered graph and then to find the disjoint components of this graph. In general, all relaxed problems can be decomposed prior to any nonsmooth computations because their structure stays fixed throughout the algorithm of Fig. 8. However, all recovery problems cannot be decomposed until they are formulated, as their structure changes as the solutions to the relaxed problems change.

#### II.4. Details and Refinements Relating to the Flowchart of Figure 8

Further explanation is provided here on how various steps of Figure 8 are solved. Note that the refinements presented here can significantly increase the speed of the relaxation procedure of step 508.

##### II.4.1 Maximization of the Non-smooth Function $\Phi_{mn}(u^{m+1}, \dots, u^n)$

One of the key steps in the Lagrangian relaxation algorithm in section II.3 is the solution of the problem

$$\text{Maximize } \{ \Phi_{mn}(u^{m+1}, \dots, u^n) : u^k \in \mathbb{R}^{M_{k+1}}; k = m+1, \dots, n \}, \quad (2.15)$$

where  $u_0^k = 0$  for all  $k = m+1, \dots, n$ . The evaluation of  $\Phi_{mn}(u^{m+1}, \dots, u^n)$  requires the optimal solution of the corresponding minimization problem (2.7). The following discussion provides some properties of these functions.

**Fact A.4.** Let  $u^{m+1}, \dots, u^n$  be multiplier vectors associated with the  $(m+1)^{\text{st}}$  through the  $n^{\text{th}}$  set of constraints (2.6), let  $\Phi_{mn}$  be as defined in (2.7), let  $V_n(\bar{z}^n)$  be the objective function value of the  $n$ -dimensional assignment problem in equation (2.6), let  $z^n$  be any feasible solution of (2.6), and let  $\bar{z}^n$  be

an optimal solution of (2.6). Then,  $\Phi_{mn}(u^{m+1}, \dots, u^n)$  is piecewise affine, concave and continuous in  $\{u^{m+1}, \dots, u^n\}$  and

$$\Phi_{mn}(u^{m+1}, \dots, u^n) \leq V_n(\bar{z}^n) \leq V_n(z^n).$$

5 (2.16)

Furthermore,

$$\Phi_{m-1n}(u^m, u^{m+1}, \dots, u^n) \leq \Phi_{mn}(u^{m+1}, \dots, u^n) \quad (2.17)$$

for all  $u^k \in \mathbb{R}^{M_{k+1}}$  with  $u_0^k = 0$  and  $k = m, \dots, n$ .

10

Most of the procedures for nonsmooth optimization are based on generalized gradients called subgradients, given by the following definition.

Definition. At  $u = (u^{m+1}, \dots, u^n)$  the set  $\partial\Phi_{mn}(u)$  is called a  
15 subdifferential of  $\Phi_{mn}$  and defined by

$$\begin{aligned} \partial\Phi_{mn}(u) = \{q \in \mathbb{R}^{M_{m+1}+1} \times \dots \times \mathbb{R}^{M_n+1} \mid \Phi_{mn}(w) - \Phi_{mn}(u) \leq q^T(w-u) \\ \forall w \in \mathbb{R}^{M_{m+1}+1} \times \dots \times \mathbb{R}^{M_n+1}\}. \end{aligned} \quad (2.18)$$

20 A vector  $q \in \partial\Phi_{mn}(u)$  is called a subgradient.

If  $z^n$  is an optimal solution of (2.7) computed during evaluation of  $\Phi_{mn}(u)$ , differentiating  $\Phi_{mn}$  with respect to  $u_{i_n}^n$  yields the following  $i_n^{th}$  component of a subgradient  $g$  of  $\Phi_{mn}(u)$

$$g_0^k = 0,$$

$$g_{i_k}^k = \sum_{i_1=0}^{N_1} \dots \sum_{i_{k-1}=0}^{N_{k-1}} \sum_{i_{k+1}=0}^{N_{k+1}} \dots \sum_{i_n=0}^{N_n} z_{i_1 \dots i_n}^n - 1 \quad (2.19)$$

for  $i_k=1, \dots, N_k$  and  $k=m+1, \dots, n$ .

5 If  $z^n$  is the unique optimal solution of (2.7),  $\partial\Phi_{mn}(u) = \{g\}$  and  $\Phi_{mn}$  is differentiable at  $u$ . If the optimal solution is not unique, then there are finitely many such solutions, say  $z^n(1), \dots, z^n(K)$ . Given the corresponding subgradients,  $g^1, \dots, g^K$ , the subdifferential  $\partial\Phi(u)$  is the convex hull of

10  $\{g^1, \dots, g^K\}$ .

#### II.4.2 Mathematical Description of a Merit or Auxiliary Function

For real-time needs, one must address the fact that the

15 non-smooth optimization problem of step 508 (Fig. 8) requires the solution of an NP-hard problem for  $m > 2$ . One approach to this problem is to use the following merit or auxiliary function to decide whether a function value has increased or decreased sufficiently in the line search or trust region

20 methods:

$$\Psi_{mn}(\bar{u}^3, \dots, \bar{u}^m; u^{m+1}, \dots, u^n) = \quad (2.20)$$

$$\begin{cases} \Phi_{mn}(u^{m+1}, \dots, u^n), & \text{if } m = 2 \\ & \text{or (2.7) is solved optimally,} \\ \Phi_{2n}(\bar{u}^3, \dots, \bar{u}^m; u^{m+1}, \dots, u^n), & \text{otherwise.} \end{cases}$$

where the multipliers  $\bar{u}^3, \dots, \bar{u}^m$  that appear in lower order  
 5 relaxations used to construct (suboptimal) solutions of the  
 $m$ -dimensional relaxed problem (2.7) have been explicitly  
 included. Note that  $\Psi_{mn}$  is well-defined since (2.9) can always  
 be solved optimally if  $m=2$ . For sufficiently small problems  
 (2.7) or (2.9), one can more efficiently solve the NP-hard  
 10 problem by branch and bound. This is the reason for the  
 inclusion of the first case; otherwise, the relaxed function  
 $\Phi_{2n}$  is used to guide the nonsmooth optimization phase. That  
 the merit function provides a lower bound for the optimal  
 solution follows directly from Fact A.4 and the following  
 15 fact.

**Fact A.5.** Given the definition of  $\Psi_{mn}$  in (2.20) above,  
 $\Psi_{mn}(\bar{u}^3, \dots, \bar{u}^m; u^{m+1}, \dots, u^n) \leq \Phi_{mn}(u^{m+1}, \dots, u^n)$  for all multipliers  
 $\bar{u}^3, \dots, \bar{u}^m, u^{m+1}, \dots, u^n$ . (2.21)

The actual function value used in the optimization phase is  $\psi_{mn}$ ; however, the subgradients are computed as in (2.19), but with the solution  $z_{i_1 \dots i_n}^n$  being a suboptimal solution constructed from a relaxation procedure applied to the  $m$ -dimensional problem. Again, the use of these lower order relaxed problems is the reason for the inclusion of the multipliers  $\bar{u}^3, \dots, \bar{u}^m$ .

To explain how the merit function is used, suppose there is a current multiplier set  $(u_{old}^{m+1}, \dots, u_{old}^n)$  and it is desirable to update to a new multiplier set  $(u_{new}^{m+1}, \dots, u_{new}^n)$  via

$$(u_{new}^{m+1}, \dots, u_{new}^n) = (u_{old}^{m+1}, \dots, u_{old}^n) + (\Delta u^{m+1}, \dots, \Delta u^n).$$

Then  $\psi_{mn}(\bar{u}_{old}^3, \dots, \bar{u}_{old}^m; u_{old}^{m+1}, \dots, u_{old}^n)$  is computed where  $(\bar{u}_{old}^3, \dots, \bar{u}_{old}^m)$  is obtained during the relaxation process used to compute a high quality solution to the relaxed  $m$ -dimensional assignment problem (2.7) at  $(u^{m+1}, \dots, u^n) = (u_{old}^{m+1}, \dots, u_{old}^n)$ . The decision to accept  $(u_{new}^{m+1}, \dots, u_{new}^n)$  is then based on  $\psi_{mn}(\bar{u}_{old}^3, \dots, \bar{u}_{old}^m; u_{old}^{m+1}, \dots, u_{old}^n) < \psi_{mn}(\bar{u}_{new}^3, \dots, \bar{u}_{new}^m; u_{new}^{m+1}, \dots, u_{new}^n)$  or some other stopping criteria commonly used in line searches. Again,  $(\bar{u}_{new}^3, \dots, \bar{u}_{new}^m)$  represents the multiplier set used in the lower level relaxation procedure to construct a high quality feasible solution to the  $m$ -dimensional relaxed problem (2.7) at  $(u^{m+1}, \dots, u^n) = (u_{new}^{m+1}, \dots, u_{new}^n)$ . The point is that each time one changes  $(u^{m+1}, \dots, u^n)$  and uses the merit function



$\psi_{mn}(\bar{u}^1, \dots, \bar{u}^m; u^{m+1}, \dots, u^n)$  for comparison purposes, one must generally change the lower level multipliers  $(\bar{u}^1, \dots, \bar{u}^m)$ .

An illustration of this merit function for  $m=n-1$  is given in (A.B. Poore and N. Rijavec. Partitioning multiple data sets: multidimensional assignments and Lagrangian relaxation. In P.M. Pardalos and H. Wolkowicz, editors, *Quadratic assignment and related problems: DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 16, pages 25-37, 1994).

#### II.4.3 Non-smooth Optimization Methods

By Fact A.4 the function  $\Phi_{mn}(u)$  is a continuous, piecewise affine, and concave, so that the negative of  $\Phi_{mn}(u)$  is convex. Thus the problem of maximizing  $\Phi_{mn}(u)$  is one of nonsmooth optimization. Since there is a large literature on such problems, only a brief discussion of the primary classes of methods for solving such problems is provided. A first class of methods, known as subgradient methods, are reviewed and analyzed in (N.Z. Shor. *Minimization Methods for Non-Differentiable Functions*. Springer-Verlag, New York, 1985). Despite their relative simplicity, subgradient methods have some drawbacks that make them inappropriate for the tracking problem. They do not guarantee a descent direction at each

iteration, they lack a clear stopping criterion, and exhibit poor convergence (less than linear).

A more recent and sophisticated class of methods are the bundle methods; excellent developments are presented in (J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I and II*. Springer-Verlag, Berlin, 1993; K.C. Kiwiel. *Methods of Descent for Non-Differentiable Optimization*. In A. Dold and B. Eckmann, editors, *Lecture Notes in Mathematics*, volume 1133, Berlin, 1985. Springer-Verlag). Bundle methods retain a set (or bundle) of previously computed subgradients to determine the best possible descent direction at the current iteration. Because of the nonsmoothness of  $\Phi_{mn}$ , the resulting direction may not provide a "sufficient" decrease in  $\Phi_{mn}$ . In this case, bundle algorithms take a "null" step, wherein the bundle is enriched by a subgradient close to  $u_k$ . As a result, bundle methods are non-ascent methods which satisfy the relaxed descent condition  $\Phi_{mn}(u_{k+1}) < \Phi_{mn}(u_k)$  if  $u_{k+1} \neq u_k$ . These methods have been shown to have good convergence properties. In particular, bundle method variants have been proven to converge in a finite number of steps for piecewise affine convex functionals in (K.C. Kiwiel. An aggregate subgradient method for non-smooth convex minimization. *Mathematical Programming*, 27:320-341,

1983; H. Schramm and J. Zowe. A version of the bundle idea for minimizing a non-smooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2, No. 1:121-152, February, 1992).

5 All of the above non-smooth optimization methods require the computation of  $\Phi_{mn}(u)$  and a  $g \in \partial\Phi_{mn}(u)$ . These in turn require the computation of the relaxed cost coefficients (2.8). In both the first and second procedures discussed in section II.3.1, maximization of  $\Phi_{2n}(u)$  must be repeatedly  
 10 evaluated. In the most efficient implementations presently known of these two procedures, it was found that at least 95% of the computational effort of the entire procedure is spent in the evaluation of the relaxed cost coefficients (2.8) as part of computing  $\Phi_{2n}(u)$ . Thus, generally a method should be  
 15 chosen that makes as efficient use of the subgradients and function values as possible, even at the cost sophisticated manipulation of the subgradients. In evaluating three different bundle procedures: (a) the conjugate subgradient method of Wolfe used in section I.1 of the first embodiment of  
 20 the present invention; (b) the aggregate subgradient method of Kiwiel (K.C. Kiwiel. An aggregate subgradient method for non-smooth convex minimization. *Mathematical Programming*, 27:320-341, 1983); and (c) the bundle trust method of Schramm and

Zowe (H. Schramm and J. Zowe. A version of the bundle idea for minimizing a non-smooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2, No. 1:121-152, February, 1992), it was determined that for a fixed number of non-smooth iterations, say, ten, the bundle-trust method provides good quality solutions with the fewest number of function and subgradient evaluations of all the methods, and is therefore the preferred method.

#### 10 II.4.4 The Two-Dimensional Assignment Problem

The forward/reverse auction algorithm of Bertsekas and Castañon (D.P. Bertsekas and D.A. Castañon. A forward/reverse auction algorithm for asymmetric assignment problems. *Computational Optimization and Applications*, 1:277-298, 1992)

15 is used to solve the many relaxed two-dimensional problems that occur in the course of execution.

#### II.4.5. Initial Multipliers and Hot Starts

20 The effective use of "hot starts" is fundamental for real-time applications. A good initial set of multipliers can significantly reduce the number of non-smooth iterations (and hence the number of  $\phi_{mn}$  evaluations) required for a high quality recovered solution. Further, there are several ways

that multipliers can be reused. First, if an  $n$ -dimensional problem is relaxed to an  $m$ -dimensional problem, relaxation provides the multiplier set  $\{u^{m+1}, u^{m+2}, \dots, u^n\}$ . These can be used as the initial multipliers for the  $(n-m+1)$ -dimensional  
 5 recovery problem for  $n-m+1 > 2$ . This approach has also worked well to reduce the number of non-smooth iterations during recovery.

Further, for track maintenance, initial feasible solutions are generated as follows. When a new scan of  
 10 information (a new observation set) arrives from a sensor, one can obtain an initial primal feasible solution by matching new reports to existing tracks via a two-dimensional assignment problem. This is known as the track-while-scan (TWS) approach. Thus, an initial primal solution exists and then we  
 15 use the above relaxation procedure to make improvements to this TWS solution. Also for track maintenance, multipliers are available from a previously solved and closely related multidimensional assignment problems for all but the new observation set.

20

#### II.4.6. Local Search Methods

Given a feasible solution of the multidimensional assignment problem, one can consider local search procedures

to improve this result, as described in (C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982; J. Pearl. *Heuristics: Intelligent Search Strategies for*  
5 *Computer Problem Solving*. Addison-Wesley, Reading, MA, 1984). One method is the idea of  $k$  interchanges. Since for sparse problems only those active arcs that participate in the solution are stored, it is difficult to efficiently identify eligible variable exchange sets with some data structures for  
10 solving the assignment problem. However, a local search that may be more promising is to use the two-dimensional assignment solver in the following way. Given a feasible solution to the multidimensional assignment problem, the indices that correspond to active arcs in the solution are enumerated.  
15 Subsequently, one coordinate is freed to formulate a two-dimensional assignment problem with one index corresponding to the enumeration and the other to the freed coordinate, and then solve a two-dimensional assignment problem to improve the freed index position.

20

#### II.4.7. Problem Decomposition

The procedures described thus far are all based on relaxation. Due to the sparsity of assignment problems,

however, frequently a decomposition of the problem into a collection of disjoint components can be done wherein each of the components can be solved independently. Due to the setup costs of Lagrangian relaxation, a branch and bound procedure

5 is generally more efficient for small components, say ten to twenty feasible 0-1 variables (i.e.,  $z_{i_1 \dots i_n}$ ). Otherwise, the relaxation procedures described above are used. Perhaps the easiest way to view the decomposition method is to view the reports or measurements as a layered graph. A vertex is

10 associated with each observation point, and an edge is allowed to connect two vertices only if the two observations belong to at least one feasible track of observations. Given this graph, the decomposition problem can then be posed as that of identifying the connected subcomponents of a graph which can

15 be accomplished by constructing a spanning forest via a depth first search algorithm, as described in (A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, MA, 1974). Additionally, decomposition of a different type might be based on the

20 identification of bi- and tri-connected components of a graph and enumerating on the connections. Here is a technical explanation. Let

$$Z = \{z_{i_1 i_2 \dots i_n} | z_{i_1 i_2 \dots i_n} \text{ is not preassigned to zero}\}$$

denote the set of assignable variables. Define an undirected graph  $G(N,A)$  where the set of nodes is

$$N = \{z_{i_n}^n | n=1, \dots, n; i_n=1, \dots, N_m\}$$

and arcs,

$$5 \quad A = \{(z_{j_n}^n, z_{j_1}^1) | n \neq 1, j_n \neq 0, j_1 \neq 0 \text{ and there exists } z_{i_1 i_2 \dots i_n} \in \mathbb{Z} \\ \text{such that } j_n = i_n \text{ and } j_1 = i_1\}.$$

Note that the nodes corresponding to zero index have not been included in the above defined graph, since two variables that have only the zero index in common can be assigned  
10 independently. Connected components of the graph are then easily found by constructing a spanning forest via a depth first search. (A detailed algorithm can be found in the book by Aho, Hopcroft and Ullman cited above). Furthermore, this procedure is used at each level in the relaxation, i.e., is  
15 applied to each assignment problem (1.4) for  $n=3, \dots, M$ .

The original relaxation problem is decomposed first. All relaxed assignment problems can be decomposed a priori and all recovery problems can be decomposed only after they are formulated. Hence, in the  $n$ -to- $(n-1)$  case, we have  $n-2$   
20 relaxed problems that can all be decomposed initially, and the recovery problems that are not decomposed (since they are all two-dimensional). In the  $n$ -to-2 case, we have only one relaxed problem that can be decomposed initially. This case



yields  $n-3$  recovery problems, which can be decomposed only after they are formulated.

5      III. NEW APPROACHES TO TRACK INITIATION AND MAINTENANCE  
         USING MULTIDIMENSIONAL ASSIGNMENT PROBLEMS

III.1. Introduction

         The ever-increasing demand for sensor surveillance systems is to accurately track and identify objects in real-time, even for dense target scenarios and in regions of high track contention. The use of multiple sensors, through more varied information, has the potential to greatly improve target state estimation and identification. However, to take full advantage of the available data, a multiple frame data association approach is needed. The most popular such approach is an enumerative technique called multiple hypothesis tracking (MHT). As an enumerative technique, MHT can be too computationally intensive for real-time needs because this problem is NP-hard. A promising approach is to utilize the recently developed Lagrangian relaxation algorithms (K.P.Pattipati, S. Deb, and Y.Bar-Shalom. A multisensor-multitarget data association algorithm for heterogeneous sensors. *IEEE Transactions on Aerospace and Electronic Systems*, 29, No. 2:560-568, April 1993; A.B. Poore and N.Rijavec. Partitioning multiple data sets:

multidimensional assignments and lagrangian relaxation, in quadratic assignment and related problems, P.M. Pardalos and H.Wolkowicz, editors, *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, 16:25-37, 1994; A.J. Robertson III. A class of lagrangian relaxation algorithms for the multidimensional problem. *Ph.D. Thesis, Colorado State University, Ft. Collins, CO*, 1995) for the multidimensional assignment problem; however, there are many other potentially good approaches to these assignment problems such as LP relaxation combined with an interior point method, GRASP, and parallelization.

These data association problems are fundamentally important in tracking. Thus, our first objective is to present an overview of the tracking problem and the context within which these data association problems fit. Following a brief review of the probabilistic framework for data association, the second objective is to formulate two models for track initiation and maintenance as multidimensional assignment problems. This formulation is based on a window moving over the frames of data. The first and simpler model uses the same window length for track initiation and maintenance, while the second model uses different lengths. As the window moves over the frames of data, one creates a

sequence of these multidimensional assignment problems and there is an overlap region of frames common to adjacent windows. Thus, given the solution of one problem, one can develop a warm start for the solution to the next problem in  
 5 the sequence, as shown hereinafter. Such information is critically important to the design of real-time algorithms.

### III.2. Overview of the Tracking Problem

Tracking and data fusion are complex subjects of  
 10 specialization that can only be briefly summarized as they are related to the subject of this paper. The processing of track multiple targets using data from one or more sensors is typically partitioned into two stages or major functional blocks, namely, signal processing and data processing (Y.Bar-  
 15 Shalom. *Multitarget-Multisensor Tracking: Advanced Applications*. Artech House, Dedham, MA, 1990; Y.Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Dedham, MA, 1986). The  
 20 first stage is the signal processing that takes raw sensor data and outputs reports. Reports are sometimes called observations, threshold exceedances, plots, hits, or returns, depending on the type of sensor. The true source of each

report is usually unknown and can be due to a target of interest, a false signal, or persistent background objects that can be moving in the field of view of the sensor.

Two principal functions of data processing are tracking  
5 and discrimination or target identification. The  
discrimination or target identification function distinguishes  
between tracks that are for targets of interest and other  
tracks such as those due to background. Also, if there is  
enough information, each track is classified as to the type of  
10 target it appears to represent. Target identification and  
discrimination will not be discussed further in this paper  
except to comment here on the use of attributes (including  
identification information) in the data association.

There are two types of attributes or features, namely,  
15 discrete valued variables and continuous valued variables.  
Available attributes of either or both types should be used in  
computing the log likelihoods or scores for data association.  
In discussing tracking in this paper, the attributes will not  
be mentioned explicitly but it should be understood that some  
20 reports and some tracks may include attributes information  
that should be and can be used in the track processing (both  
data association and filtering) if it is useful.

III.2.1. Tracking Functions

The tracking function accepts reports for each frame of data and constructs or maintains a target state estimate, the variance-covariance matrix of the state estimation error, and refined estimates (or probabilities) of the target attributes. The state estimate typically includes estimates of target position and velocity in three dimensions and possibly also acceleration and other variables of interest.

The tracking function is typically viewed in two stages, namely, data association and filtering. Also, the process of constructing new tracks, called track initiation, is different from the process of updating existing tracks, called track maintenance.

In track maintenance, the data association function decides how to relate the reports from the current frame of data to the previously computed tracks. In one approach, at most one report is assigned to each track, and in other approaches, weights are assigned to the pairings of reports to a track. After the data association, the filter function updates each target state estimate using the one or more (with weights) reports that were determined by the data association function. A filter commonly used for multiple target tracking and data fusion is the well known Kalman filter (or the

extended Kalman filter in the case of nonlinear problems) or a simplified version of it (Y. Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications*. Artech House, MA, 1990; Y.Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Dedham, MA, 1986).

10 In track initiation, typically a sequence of reports is selected (one from each of a few frames of data) to construct a new track. In track initiation, the filtering function constructs a target state estimate and related information based on the selected sequence of reports. The new track is later updated by the track maintenance processing of the subsequent frames of reports.

15 In some trackers, there is also a tentative tracking function for processing recently established tracks until there is enough confidence to include them in track maintenance. While for simplicity of presentation, tentative tracking will not be included in the processing discussed in  
20 this paper, the techniques discussed could readily include one or more tentative tracking functions.

There have been numerous approaches developed to perform the data association function. Since optimal data association

is far too complex to implement, good but practical sub-optimal approaches are pursued. Data association approaches can be classified in a number of ways. One way to classify data association approaches is based on the number of data frames used in the association process (O.E. Drummond. Multiple sensor tracking with multiple frame, probabilistic data association. In *Signal and Data Processing of Small Targets*, SPIE Proceedings, volume 2561, pages 322-336, 1995).

In single frame data association for track maintenance, "hard decisions" are made on the assignment of reports to the tracks. Some single frame approaches include: individual nearest neighbor, PDA, JPDA, and global nearest neighbor (sequential most probable hypothesis tracking), which uses a two-dimensional assignment algorithm (Y. Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications*. Artech House, MA, 1990; Y. Bar-Shalom and T.E. Fortmann. *Tracking and Data Association*. Academic Press, Boston, MA, 1988; S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Dedham, MA, 1986). In most single frame data association approaches, only one track per object is carried forward to be used in processing the next frame of reports.

Multiple frame data association is more complex and frequently involves purposely carrying forward more than one track per target to be used in processing the next frame of reports. By retaining more than one track per target, the tracking performance is improved at the cost of increased processing load. The best known multiple frame data association approach is an enumerative technique called multiple hypothesis tracking (MHT) which enumerates all the global hypothesis with various pruning rules.

As shown hereinafter, the log likelihood of the probability of a global hypothesis can be decomposed into the sum of scores of each track that is contained in the hypothesis. As a consequence, the most probable hypothesis can be identified with the aid of a non-enumerative algorithm for the assignment so formulated.

#### III.2.2. Interface Between Track Initiation And Track Maintenance

There are a number of ways that track initiation and track maintenance functions can interact (O.E. Drummond. Multiple target tracking with multiple frame, probabilistic data association. In *Signal and Data Proceedings*, volume 1954, pages 394-408, 1993). Two ways that are especially pertinent to the methods of this paper will be discussed in this section.



One approach is to treat these two functions sequentially, by first assigning reports to tracks and then using the information that remains to form new tracks. A better approach is to integrate both processes where  
5 initiating tracks compete for the new frame of reports on an equal basis, i.e., at the same time.

In the integrated approach, the data association for track initiation and track maintenance processing are combined and conducted simultaneously. One assignment array is created  
10 that includes the track scores for potential tracks for both track initiation and track maintenance. The first dimension of the assignment problem includes only all the tracks either created or updated in the processing of the prior frames of reports. Each of the remaining dimensions accommodates one  
15 frame of reports.

After the assignment algorithm finds a solution, each of the previously established tracks are updated with the report assigned to it from the second dimension of the assignment array. The remaining reports in the second dimension that  
20 were in the assignment solution are firmly established as track initiators. The unassigned reports in the second dimension of the assignment array are discarded as false signals. The processing of the next frame of reports repeats

this process using these updated tracks and the newly identified initiators in the first dimension of the cost array for processing the new frame of reports. Details of this approach are discussed hereinafter.

5       The integrated approach just discussed, uses the same number of frames of reports for both track initiation and track maintenance. The goal in using the multi-dimensional assignment algorithm is to provide improved performance while minimizing the amount of processing required. Typically,  
10 track initiation will benefit from more frames of reports than will track maintenance. Thus, a second approach that integrates track initiation and track maintenance is discussed hereinafter, wherein the number of frames of reports is not the same for these two functions. This is a novel approach  
15 that is introduced in this paper.

### III.2.3. Multiple Sensor Processing

There are many advantages and many ways of combining data from multiple sensors. There are also many ways of  
20 categorizing the different algorithmic architectures for processing data from multiple sensors. One approach outlines four generic algorithmic architectures (O.E. Drummond. Multiple sensor tracking with multiple frame, probabilistic

data association. In *Signal and Data Processing of Small Targets*, SPIE Proceedings, volume 2561, pages 322-336, 1995). Two of these generic architectures are especially pertinent to this paper and are summarized briefly.

5        In the Centralized Fusion algorithmic architecture, reports are combined from the various sensors to form global tracks. This algorithmic architecture is also called Central Level Tracking, Centralized Algorithmic Architecture, or simply the Type IV algorithmic architecture. In track  
10 maintenance, for example, if a single frame data association is used, then for data association a frame of reports from one sensor is processed with the latest global tracks; then the global tracks are updated by the filter function. After the processing of this frame of reports is completed, a frame of  
15 the reports from another (or the same) sensor is processed with these updated global tracks. This process is continued as new frames of data become available to the system as a whole.

      In Centralized Fusion, using the multi-dimensional  
20 assignment algorithm for the data association with multiple sensors is similar to the processing of data from a single sensor. Instead of using multiple frames of reports from a single sensor, the multiple frames of reports come from

multiple sensors. The frames of reports from all the sensors are ordered based on the nominal time each frame of reports is obtained and independent of which sensor provided the reports. In this way the approaches discussed in this paper can be  
5 extended to process reports from multiple sensors using the Centralized Fusion algorithmic architecture.

The second pertinent algorithmic architecture is Track Fusion. This approach is also called the Hierarchical or Federated algorithmic architecture, sensor level tracking or  
10 simply the Type II algorithmic architecture. In track maintenance, for example, a processor for each sensor computes single sensor tracks; these tracks are then forwarded to the global tracker to compute global tracks based on data from all sensors.

15 After the first time a sensor processor forwards tracks to the global tracker, then subsequent tracks for the same targets are cross-correlated with the existing global tracks. This track-to-track cross-correlation is due to the common history of the current sensor tracks and the tracks from the  
20 same sensor that were forwarded earlier to the global tracker. The processing must take this cross-correlation into account and there are a number of ways of compensating for this-cross-correlations. One method for dealing with this cross-

correlation is to decorrelate the sensor tracks that are sent to the global tracker. There are a variety of ways to achieve this decorrelation and some are summarized in recent paper (O.E. Drummond. Feedback in track fusion without process  
5 noise. In *Signal and Data Processing of Small Targets, SPIE Proceedings, volume 2561, pages 369-383, 1995*).

Once the sensor tracks are decorrelated they can be processed by the global tracker in almost the same way as reports are processed. In the case of track fusion the  
10 association process is referred to as track (or track-to-track) association rather than data (or report-to-track) association. If the sensor tracks are decorrelated, the global tracker can process the tracks from the various sensor processors in much the same way that the global tracker of  
15 Centralized Fusion processes reports. Accordingly the methods described in this paper can be readily extended to processing data from multiple sensors using either the Centralized Fusion or Track Fusion or even a hybrid combination of both algorithmic architectures.

20

### III.3. Formulation of the Data Association Problem

The goal of this section is to briefly outline the probabilistic framework for the data association problems

presented in this work. The technical details are presented elsewhere (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994). The data association problems for multisensor and multitarget tracking considered in this work are generally posed (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) as that of maximizing the posterior probability of the surveillance region (given the data) according to

$$\text{Maximize } \{P(\Gamma=\gamma | Z^{N+1}) | \gamma \in \Gamma^*\}, \quad (3.1)$$

where  $Z^{N+1}$  represents  $N+1$  data sets,  $\gamma$  is a partition of indices of the data (and thus induces a partition of the data),  $\Gamma^*$  is the finite collection of all such partitions,  $\Gamma$  is a discrete random element defined on  $\Gamma^*$ , and  $P(\Gamma=\gamma | Z^{N+1})$  is the posterior probability of a partition  $\gamma$  being true given the data  $Z^{N+1}$ . The term partition is defined below; however, this framework is currently sufficiently general to cover set packings and coverings.

Consider  $N+1$  data sets  $Z(k)$  ( $k=1, \dots, N+1$ ), each consisting of  $M_k$  actual reports and a dummy report  $z_0^k$ , and let denote the cumulative data set defined by

$$Z(k) = \left\{ z_{i_k}^k \right\}_{i_k=0}^{M_k} \text{ and } Z^{N+1} = \{Z(1), \dots, Z(N+1)\}, \quad (3.2)$$

respectively. (The dummy report  $z_0^k$  serves several purposes in the representation of missing data, false reports, initiating tracks, and terminating tracks (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).) In multisensor data fusion and multitarget tracking the data sets  $Z(k)$  may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are reports that must be partitioned into tracks and false alarms. In our formulation of track maintenance, which uses a moving window, one data set will be tracks and remaining data sets will be reports which are assigned to existing tracks, as false reports, or to initiating tracks.

We specialize the problem to the case of set partitioning (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and

multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) defined in the following way. Define a "track of data" as  $\{z_{i_1}^1, \dots, z_{i_{N+1}}^{N+1}\}$  where each  $i_k$  can assume zero or nonzero values. A partition of the data will  
 5 refer to a collection of tracks of data wherein each report occurs exactly once in one of the tracks of data and such that all data is used up; the occurrence of dummy report is unrestricted. The reference partition  $\gamma^0$  is that in which all reports are declared to be false.

10 Next, under appropriate independence assumptions one can show (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994)

15

$$\frac{P(\Gamma = \gamma | Z^{N+1})}{P(\Gamma = \gamma^0 | Z^{N+1})} \equiv L\gamma \equiv \prod_{(i_1 \dots i_{N+1}) \in \gamma} L_{i_1 \dots i_{N+1}}, \quad (3.3)$$

$L_{i_1 \dots i_{N+1}}$  is a likelihood ratio containing probabilities for  
 20 detection, maneuvers, and termination as well as probability density functions for report errors, track initiation and termination. Define

25

$$\begin{aligned} c_{i_1 \dots i_{N+1}} &= -\ln L_{i_1 \dots i_{N+1}}, \\ z_{i_1 \dots i_{N+1}} &= \begin{cases} 1, & \text{if } (z_{i_1 \dots i_{N+1}}) \text{ are assigned to as a track,} \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \quad (3.4)$$



Then, recognizing that  $-\ln \left[ \frac{P(\Gamma = \gamma | Z^{N+1})}{P(\Gamma = \gamma^0 | Z^{N+1})} \right] = \sum_{(i_1, \dots, i_{N+1}) \in \gamma} c_{i_1 \dots i_{N+1}}$ , the problem (3.1) can be expressed as the  $(N+1)$ -dimensional assignment problem

$$\begin{aligned}
 & \text{Minimize} \quad \sum_{i_1=0}^{M_1} \dots \sum_{i_{N+1}=0}^{M_{N+1}} c_{i_1 \dots i_{N+1}} z_{i_1 \dots i_{N+1}} \\
 & \text{Subject To} \quad \sum_{i_2=0}^{M_2} \dots \sum_{i_{N+1}=0}^{M_{N+1}} z_{i_1 \dots i_{N+1}} = 1, \quad i_1 = 1, \dots, M_1, \\
 & \quad \sum_{i_1=0}^{M_1} \dots \sum_{i_{k-1}=0}^{M_{k-1}} \sum_{i_{k+1}=0}^{M_{k+1}} \dots \sum_{i_{N+1}=0}^{M_{N+1}} z_{i_1 \dots i_{N+1}} = 1 \\
 & \quad \text{for } i_k = 1, \dots, M_k \text{ and } k = 2, \dots, N, \\
 & \quad \sum_{i_1=0}^{M_1} \dots \sum_{i_{N+1}=0}^{M_{N+1}} z_{i_1 \dots i_{N+1}} = 1, \quad i_{N+1} = 1, \dots, M_{N+1}, \\
 & \quad z_{i_1 \dots i_{N+1}} \in \{0, 1\} \text{ for all } i_1, \dots, i_{N+1},
 \end{aligned} \tag{3.5}$$

- 5 where  $c_{0 \dots 0}$  is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each non-dummy report occurs exactly once in a "track of data." One can modify this formulation to include multi-assignments of one, some, or all the actual reports. The assignment problem
- 10 (3.5) is changed accordingly. For example, if  $z_{i_k}^k$  is to be assigned no more than, exactly, or no less than  $n_{i_k}^k$  times, then the "=1" in the constraint (3.5) is changed to " $\leq$ ", " $=$ ", " $\geq$ "  $n_{i_k}^k$ ,"

respectively. In making these changes, one must pay careful attention to the independence assumptions, which need not be valid in many applications. Expressions for the likelihood ratios can be found in the work of (A.B. Poore.

- 5 Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995) and the references therein.

10

#### III.4. Track Initiation and Maintenance

- In this section we explain two multiframe assignment formulations to the track initiation and maintenance problem. The continued use of all prior information is computationally
- 15 intensive for tracking, so that a window sliding over the frames of reports is used as the framework for track maintenance and track initiation within the window. The objectives are to describe an improved version of a simple method and then to put this into a more general framework in
- 20 which track initiation and maintenance have different length moving windows.

### III.4.1. The First Approach to Track Maintenance and Initiation

The first method as explained in this section is an improved version of our first track maintenance scheme (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994) and uses the same window length for track initiation and maintenance after the initialization step. The process is to start with a window of length  $N+1$  anchored at frame one. In the first step there is only one track initiation in that we assume no prior existing tracks. In the second and all subsequent frames, there is a window of length  $N$  anchored at frame  $k$  plus a collection of tracks up to frame  $k$ . This window is denoted by  $\{k, k+1, \dots, k+N\}$ . The following explanation of the steps is much like mathematical induction in that we explain the first step and then step  $k$  to step  $k+1$ .

**Track Maintenance and Initiation: Step 1.** Let

$$\{i_1(l_2), i_2(l_2), \dots, i_N(l_2), i_{N+1}(l_2)\}_{l_2=1}^{L_2} \quad (3.6)$$

be an enumeration of all those zero-one variables in the solution of the assignment problem (3.5) (i.e.,  $z_{i_1 i_2 \dots i_{N+1}} = 1$ ) excluding all the false reports in the solution (i.e., all

those zero-one variables with exactly one nonzero index) and zero-one variables in the solution for which  $(i_1, i_2) = (0, 0)$ . (The latter can correspond to tracks that initiate on frames three and higher.) These denote our initial tracks.

5 Consider only the first two index sets in this enumeration (3.6) and add the zero index  $l_2=0$  with the corresponding values of  $i_1$  and  $i_2$  being zero. Thus, the enumeration is now  $\{i_1(l_2), i_2(l_2)\}_{l_2=0}^{L_2}$ . The notation  $T_2(l_2) \equiv (z_{i_1(l_2)}, z_{i_2(l_2)})$  will be used for this pairing. Suppose  
 10 now that the next data set, i.e., the  $(N+2)^{\text{th}}$  set, is added to the problem.

To explain the costs for the new problem, one starts with the hypothesis that a partition  $\gamma \in \Gamma^*$  being true is now conditioned on the truth of the pairings on the first two  
 15 frames being correct. Corresponding to the sequence  $\{T_2(l_2), z_{i_3}, \dots, z_{i_{N+2}}\}$ , the likelihood function is then given by

$$L_\gamma \equiv \prod_{\{T_2(l_2), z_{i_3}, \dots, z_{i_{N+2}}\} \in \gamma} L_{l_2 i_3 \dots i_{N+2}}, \text{ where} \quad (3.7)$$

$$L_{l_2 i_3 \dots i_{N+2}} \equiv L_{T_2(l_2)} L_{z_{i_3}, \dots, z_{i_{N+2}}},$$

$$L_{T_2(0)} = 1.$$

Next, define the cost and the corresponding zero-one variable

20

$$C_{l_2 i_3 \dots i_{N+2}} = -\ln L_{l_2 i_3 \dots i_{N+2}},$$

$$z_{l_2 i_3 \dots i_{N+2}} = \begin{cases} 1, & \text{if } \{z_{i_3}^3, \dots, z_{i_{N+2}}^{N+2}\} \text{ is assigned to Track } T_2(l_2), \\ 0, & \text{otherwise,} \end{cases} \quad (3.8)$$

respectively. Then the track maintenance problem Maximize  $\{L_\gamma | \gamma \in \Gamma^*\}$  can be formulated as the following multidimensional assignment

$$\begin{aligned} 10 \quad & \text{Minimize} \quad \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} \dots \sum_{i_{N+2}=0}^{M_{N+2}} C_{l_2 i_3 \dots i_{N+2}} z_{l_2 i_3 \dots i_{N+2}} \\ & \text{Subject to:} \quad \sum_{i_3=0}^{M_3} \dots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_2 i_3 \dots i_{N+2}} = 1, \quad l_2 = 1, \dots, L_2, \\ & \quad \sum_{l_2=0}^{L_2} \sum_{i_4=0}^{M_4} \dots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_2 i_3 \dots i_{N+2}} = 1, \quad i_3 = 1, \dots, M_3, \\ & \quad \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} \dots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}} \dots \sum_{i_{N+2}=0}^{M_{N+2}} z_{l_2 i_3 \dots i_{N+2}} = 1, \\ & \quad \text{for } i_p = 1, \dots, M_p \text{ and } p = 4, \dots, N+2-1, \\ & \quad \sum_{i_3=0}^{M_3} \dots \sum_{i_{N+2-1}=0}^{M_{N+2-1}} z_{l_2 i_3 \dots i_{N+2}} = 1, \quad i_{N+2} = 1, \dots, M_{N+2} \\ & \quad z_{l_2 i_3 \dots i_{N+2}} \in \{0, 1\} \text{ for all } l_2, i_3, \dots, i_{N+2} \end{aligned} \quad (3.9)$$

**Track Maintenance and Initiation: Step k.** At the beginning of the  $k^{th}$  step, we solve the following (N+1)-  
15 dimensional assignment problem.

$$\text{Minimize} \quad \sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \dots \sum_{i_{k+N}=0}^{M_{k+N}} c_{l_k i_{k+1} \dots i_{k+N}} z_{l_k i_{k+1} \dots i_{k+N}} \quad (3.10)$$

$$\begin{aligned} \text{Subject to:} \quad & \sum_{i_{k+1}=0}^{M_{k+1}} \dots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \dots i_{k+N}} = 1, \quad l_k = 1, \dots, L_k, \\ & \sum_{l_k=0}^{L_k} \sum_{i_{k+2}=0}^{M_{k+2}} \dots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \dots i_{k+N}} = 1, \quad i_{k+1} = 1, \dots, M_{k+1}, \\ & \sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \dots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}} \dots \sum_{i_{k+N}=0}^{M_{k+N}} z_{l_k i_{k+1} \dots i_{k+N}} = 1 \\ & \text{for } i_p = 1, \dots, M_p \text{ and } p = k+2, \dots, N+k-1, \end{aligned}$$

$$\begin{aligned} & \sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} \dots \sum_{i_{k+1+N-2}=0}^{M_{k+1+N-2}} z_{l_k i_{k+1} \dots i_{k+N}} = 1, \quad i_{k+N} = 1, \dots, M_{k+N}, \\ & z_{l_k i_{k+1} \dots i_{k+N}} \in \{0, 1\} \text{ for all } l_k, i_{k+1}, \dots, i_{k+N}, \end{aligned}$$

5

where for the first step  $l_1$  and  $L_1$  are replaced by  $i_1$  and  $M_1$ , respectively. The first index  $l_k$  in the subscripts corresponds to the sequence of tracks  $\{T_k(l_k)\}_{l_k=0}^{L_k}$ , where  $T_k(l_k) = \{z_{i_1}^1(l_k), \dots, z_{i_k}^k(l_k)\}$  is a track of data from the solution of the problem prior to the formulation of (3.10). If  $k=1$ , then it is just the first data set in frame one.

**Basic Assumption.** Suppose problem (3.10) has been solved and let the solution, i.e., those zero-one variables equal to one, be enumerated by

$$\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}), \dots, i_{k+N}(l_{k+1}))\}_{l_{k+1}=1}^{L_{k+1}} \quad (3.11)$$

with the following exceptions.

15

- a. All zero-one variables for which  $(l_k, i_{k+1}) = (0, 0)$  are excluded. Thus, tracks that initiate on frames after the  $(k+1)^{\text{th}}$  are not included in the list.
- b. All zero-one variables whose subscripts have the form  $l_k = 0$  and exactly one nonzero index in the remaining indices  $\{i_{k+1}, \dots, i_{k+N}\}$  are excluded. These correspond to false reports on frames  $p = k+1, \dots, k+N$ .
- c. All variables  $z_{l_k i_{k+1} \dots i_{k+N}}$  for which  $(i_{k+1}, \dots, i_{k+N}) = (0, 0, \dots, 0)$  and  $i_k(l_k) = 0$  are excluded from (1.11). In other words, the reports on the last  $N+1$  frames in the  $\{T_k(l_k), z_{i_{k+1}}^{k+N}, \dots, z_{i_{k+N}}^{k+N}\}$  are all dummy. Any solution with this feature corresponds to a terminated track.

Given the enumeration (3.11), one now fixes the assignments on the first two index sets in the list (3.11). The zero index  $l_{k+1} = 0$  is added to the enumeration to specify  $(l_k(0), i_{k+1}(0)) = (0, 0)$  that is used to represent false reports and tracks that initiate on frame  $k+2$  or later, so that the enumeration (3.11) is now  $\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\}_{l_{k+1}=0}^{L_{k+1}}$ .

Then, for  $l_{k+1} = 1, \dots, L_{k+1}$ , the  $l_{k+1}^{\text{th}}$  such track is denoted by  $T_{k+1}(l_{k+1}) = \{T_k(l_k(l_{k+1})), z_{i_{k+1}}^{k+1}(l_{k+1})\}$  and the  $(N+1)$ -tuple  $\{T_{k+1}(l_{k+1}), z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\}$  will denote a track  $T_{k+1}(l_{k+1})$  plus a set of reports  $\{z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\}$ , actual or dummy, that are feasible with the track  $T_{k+1}(l_{k+1})$ . The  $(N+1)$ -tuple  $\{T_{k+1}(0), z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\}$

will denote a track that initiates in the sliding window, i.e., on subsequent frames. A false report in the sliding window is one with all but one non-zero index  $i_p$  for some  $p=k+2, \dots, k+1+N$  in the  $(N+1)$ -tuple  $\{T_{k+1}(0), z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\}$ .

- 5 The corresponding hypothesis about a partition  $\gamma \in \Gamma^*$  being true is now conditioned on the truth of the  $L_{k+1}$  tracks existing at the beginning of the N-frame window. (Thus the assignments prior to this sliding window are fixed.) The likelihood function is given by

$$L_Y = \prod_{\{T_{k+1}(l_{k+1}), z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\} \in Y} L_{l_{k+1} i_{k+2} \dots i_{k+1+N}}, \text{ where}$$

$$10 \quad \begin{aligned} L_{l_{k+1} i_{k+2} \dots i_{k+1+N}} &= L_{T_{k+1}(l_{k+1})} L_{z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}}, \\ L_{T_{k+1}(0)} &= 1. \end{aligned} \quad (3.12)$$

Next, define the cost and the zero-one variable by

$$\begin{aligned} C_{l_{k+1} i_{k+2} \dots i_{k+1+N}} &= -\ln L_{l_{k+1} i_{k+2} \dots i_{k+1+N}} \equiv -\ln L_{T_{k+1}(l_{k+1}) z_{i_{k+2}}^{k+2} \dots z_{i_{k+1+N}}^{k+1+N}}, \\ Z_{l_{k+1} i_{k+2} \dots i_{k+1+N}} &= \begin{cases} 1, & \text{if } \{z_{i_{k+2}}^{k+2}, \dots, z_{i_{k+1+N}}^{k+1+N}\} \text{ is assigned to } T_{k+1}(l_{k+1}), \\ 0, & \text{otherwise,} \end{cases} \end{aligned} \quad (3.13)$$

- respectively, so that the track extension problem, which was  
15 originally formulated as *Maximize*  $\{L_Y | Y \in \Gamma^*\}$ , can be expressed



as exactly the same multi-dimensional assignment in (3.4) with  $k$  replaced by  $k+1$ . Thus, we do not repeat it here.

#### III.4.2. The Second Approach to Track Maintenance and Initiation

In the first approach the window lengths were the same for both track maintenance and initiation. This can be inefficient in that one can usually use a shorter window for track maintenance than for track initiation. This section addresses such a formulation.

**The General Step  $k$ .** To formulate a problem for track initiation and maintenance we consider a moving window centered as frame  $k$  of length  $I+J+1$  denoted by  $[k-I, \dots, k, \dots, k+J]$ . In this representation, the window length for track maintenance is  $J$  and that for initiation,  $I+J+1$ . The objective will be to explain the situation at this center and then the move to the same length window at center  $k+1$  denoted by  $[k+1-I, \dots, k+1, \dots, k+1+J]$ , i.e., by moving the window to the right one frame at time. The explanation from the first step follows hereinafter.

The notation for a track of data is

$$T_k(l_k) = \left\{ z_{i_1(l_k)}^1, \dots, z_{i_p(l_k)}^p, \dots, z_{i_k(l_k)}^k \right\}, \quad (3.14)$$

where the index  $l_k$  is used for an enumeration of those reports paired together. We also use the notation  $T_{p,k}(l_k)$  to denote the sequence of reports belonging to track  $T_k(l_k)$  but restricted to frames prior to and including  $p$ . Thus,

5

$$\begin{aligned} T_k(l_k) &\equiv T_{k,k}(l_k), \\ T_{p,k}(l_k) &= \left\{ z_{i_1(l_k)}^1, \dots, z_{i_p(l_k)}^p \right\}, \\ T_{k,k}(l_k) &= T_{p,k}(l_k) \cup \left\{ z_{i_{p+1}(l_k)}^{p+1}, \dots, z_{i_{k-1}(l_k)}^{k-1}, z_{i_k(l_k)}^k \right\} \text{ for any } p \leq k-1. \end{aligned} \quad (3.15)$$

Given this notation for the tracks and partition of the data in the frames  $\{k-I, \dots, k, \dots, k+J\}$ ,  $L_{T_{p,k}(l_k)}$  will denote the accumulated likelihood ratio up to and including frame  $p$  ( $p \leq k$ ) for a track that is declared as existing on frame  $k$  as a solution of the assignment problem. In this notation, the likelihood for  $T_{p,k}(l_k)$  and that of the association of

10

$\left\{ z_{i_{k+1}}^{k+1}, \dots, z_{i_{k+N}}^{k+N} \right\}$  with track  $T_k(l_k)$  is given by

$$\begin{aligned} L_{T_k(l_k)} &= L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \dots i_k(l_k)} \text{ for any } p \leq k-1, \\ L_{T_k(l_k) i_{k+1} \dots i_{k+N}} &= L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \dots i_k(l_k) i_{k+1} \dots i_{k+N}} \text{ for any } p \leq k-1, \end{aligned} \quad (3.16)$$

15 respectively.

The cost for the assignment of  $\{z_{i_{k+1}}^{k+1}, \dots, z_{i_{k+N}}^{k+N}\}$  to track  $T_k(l_k)$  and the corresponding zero-one variable are given by

$$\begin{aligned}
 c_{l_k i_{k+1} \dots i_{k+J}}^T &= -\ln[L_{T_k(l_k)} L_{i_{k+1} \dots i_{k+J}}] \\
 &= -\ln[L_{T_{p,k}(l_k)} L_{i_{p+1}(l_k) \dots i_k(l_k) i_{k+1} \dots i_{k+J}}] \quad \text{and} \\
 z_{l_k i_{k+1} \dots i_{k+J}}^T &= \begin{cases} 1, & \text{if } \{z_{i_{k+1}}^{k+1}, \dots, z_{i_{k+N}}^{k+N}\} \text{ is assigned to track } T_k(l_k), \\ 0, & \text{otherwise,} \end{cases}
 \end{aligned} \tag{3.17}$$

respectively. Likewise, for costs associated with the false reports on the frames  $k-I$  to  $k$  and as associated with  $\{z_{i_{k+1}}^{k+1}, \dots, z_{i_{k+J}}^{k+J}\}$  and the corresponding zero-one variables are given by:

$$\begin{aligned}
 c_{i_{k-1} \dots i_k i_{k+1} \dots i_{k+J}}^F &= -\ln[L_{i_{k-1} \dots i_k i_{k+1} \dots i_{k+J}}] \\
 z_{i_{k-1} \dots i_k i_{k+1} \dots i_{k+J}}^F &= \begin{cases} 1, & \text{if } \{z_{i_{k-I}}^{k+1}, \dots, z_{i_{k+1}}^{k+J}, \dots, z_{i_{k+J}}^{k+J}\} \\ & \text{are assigned as a track,} \\ 0, & \text{otherwise.} \end{cases}
 \end{aligned} \tag{3.18}$$

For the window of frames in the range  $\{k-I, \dots, k\}$ , the easiest explanation of the partition of the reports is based on the definitions

$$T_{p,k} = \left\{ i_p | z_{i_p}^P \text{ belongs to one of the tracks} \right. \\ \left. \text{listed on frame } k, \text{ i.e., to one} \right. \\ \left. \text{of the } T_k(l_k) \text{ for some } l_k = 1, \dots, L_k \right\}, \quad (3.19)$$

$$F_{p,k} = (\{1, \dots, M_p\} \setminus T_{p,k}) \cup \{0\},$$

so that all of those reports not used in the tracks  $T_k(l_k)$  on frame  $p$  are put into the set of available reports  $F_{p,k}$  for the formation of tracks over the entire window. Thus, we formulate the assignment problem as

$$\begin{aligned} \text{Minimize } & \sum_{p=k-I}^k \sum_{i_p \in F_{p,k}} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} C_{i_{k-I} \dots i_k i_{k+1} \dots i_{k+J}}^F Z_{i_{k-I} \dots i_k i_{k+1} \dots i_{k+J}}^F \\ & + \sum_{l_k=1}^{L_k} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} C_{l_k i_{k+1} \dots i_{k+J}}^T Z_{l_k i_{k+1} \dots i_{k+J}}^T \\ \text{Subject To } & \sum_{\{p=k-I, p \neq q\}}^k \sum_{i_p \in F_{p,k}} \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} Z_{i_{k-I} \dots i_k i_{k+1} \dots i_{k+J}}^F = 1 \\ & \text{for } i_q \in F_{p,k} \setminus \{0\} \text{ and } q = k-I, \dots, k, \\ & \sum_{p=k-I}^k \sum_{i_p \in F_{p,k}} \sum_{\{r=k+1, r \neq q\}} \sum_{i_q=0}^{M_q} Z_{i_{k-I} \dots i_k i_{k+1} \dots i_{k+J}}^F = 1 \\ & \text{for } i_q = 1, \dots, M_q \text{ and } q = k+1, \dots, k+J, \\ & Z_{i_{k-I} \dots i_k i_{k+1} \dots i_{k+J}} \in \{0, 1\} \text{ for all } i_{k-I}, \dots, i_k, i_{k+1}, \dots, i_{k+J}, \\ & \sum_{r=k+1}^{k+J} \sum_{i_r=0}^{M_r} Z_{l_k i_{k+1} \dots i_{k+J}}^T = 1, \quad l_k = 1, \dots, L_k, \\ & \sum_{l_k=1}^{L_k} \sum_{\{r=k+1, r \neq q\}} \sum_{i_r=0}^{M_r} Z_{l_k i_{k+1} \dots i_{k+J}}^T = 1, \quad i_q = 1, \dots, M_q \\ & \text{for } q = k+1, \dots, k+J, \\ & Z_{l_k i_{k+1} \dots i_{k+J}} \in \{0, 1\} \text{ for all } l_k \geq 1, i_{k+1}, \dots, i_{k+J}. \end{aligned} \quad (3.20)$$

**Basic Assumptions.** Suppose problem (1.20) has been solved and let the solution, i.e. those zero-one variables equal to one, be enumerated by

$$5 \quad \left\{ Z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) \dots i_{k+J}(l_{k+1})}^T \right\}_{l_{k+1}=1}^{\tilde{L}_{k+1}}, \quad (3.21)$$

$$\left\{ Z_{i_{k-I}(l_{k+1}) \dots i_k(l_{k+1}) i_{k+1}(l_{k+1}) \dots i_{k+J}(l_{k+1})}^F \right\}_{l_{k+1}=\tilde{L}_{k+1}+1}^{L_{k+1}}$$

with the following exceptions.

- a. All zero-one variables in the second list for which  $(i_{k-I}, \dots, i_k, i_{k+1}) = (0, \dots, 0, 0)$  are excluded. Thus, tracks that  
 10 initiate on frames after the  $(k+1)^{th}$  are not included in the list.
- b. All false reports are excluded, i.e., all zero-one variables in the second list whose subscripts have exactly one nonzero index.
- 15 c. All variables  $Z_{l_k i_{k+1} \dots i_{k+N}}^T$  for which  $(i_{k+1}, \dots, i_{k+N}) = (0, 0, \dots, 0)$  and  $Z(p) \cap T_k(l_k) = 0$  for  $p = k-K, \dots, k$  where  $K \geq 0$  is user specified. Thus the track  $T_k(l_k)$  is terminated if it is not observed over  $K+J+1$  frames.

Given the enumeration (3.20), one now fixes the assignments on  
 20 the all index sets up to and including the  $(k+1)^{th}$  index sets.

$$T_{k+1}(l_{k+1}) = \begin{cases} \left( T_k(l_k(l_{k+1})), z_{i_{k+1}}^{k+1}(l_{k+1}) \right) \\ \text{for } l_{k+1} = 1, \dots, \tilde{L}_{k+1}, \\ \left( z_{i_{k-I}}^{k-I}(l_{k+1}), \dots, z_{l_k}^k(l_{k+1}), z_{i_{k+1}}^{k+1}(l_{k+1}) \right) \\ \text{for } l_{k+1} = \tilde{L}_{k+1} + 1, \dots, L_{k+1}. \end{cases} \quad (3.22)$$

Thus one can now formulate the assignment problem for the next problem exactly as in (3.20) but with  $k$  replaced by  $k+1$ . Thus, we do not repeat it here.

**The Initial Step.** Here is one explanation for the  
 5 initial step. First, assume that  $N=I+J$ . In this case, we start the track initiation with a solution of (3.5). Let  

$$\left\{ i_1(l_2), i_2(l_2), \dots, i_N(l_2), i_{N+1}(l_2) \right\}_{l_2=0}^{L_2}$$
 be an enumeration of the solution set of (3.5), i.e., those zero-one variables  $z_{i_1(l_2) i_2(l_2) \dots i_{N+1}(l_2)} = 1$ , including  $z_{00 \dots 0} = 1$   
 10 corresponding to  $l_2=0$ , but excluding all those zero-one variables that are assigned to one and correspond to false reports (i.e., there is exactly one nonzero index in the subscript of  $z_{i_1 i_2 \dots i_{N+1}}$ ), all those zero-one variables that are assigned to one and correspond to tracks that initiate on  
 15 frames higher than  $I+2$ . Then we fix the data association decisions corresponding to the reports in our list of tracks prior to and including frame  $k+1=I+2$ . This defines the  $k$  for

problem (3.5) and one can then continue the development by adding a frame to the window as in the general case.

If  $I+J > N$ , then one possibility is to start the process with  $N+1$  frames, and assuming  $J \leq N$ , proceed as before replacing  
 5  $I$  by  $N-J$  for the moment, and continue to add frames without lopping off the first frame in the window until reaches a window of length  $I+J+1$ . Then we proceed as in the previous paragraph.

If  $I+J < N$ , then one can solve the track initiation  
 10 problem (3.5), formulate the problem with the center of the window at  $k+1 = N+1-J$ , enumerate the solutions as above, and lop off the first  $N-J-I$  frames. Then, we proceed just as in the case  $I+J=N$ .

### III.5. CONCLUDING COMMENTS

15 A primary objective in this work has been to demonstrate how multidimensional assignment problems arise in the tracking environment. The problem of track initiation and maintenance has been formulated within the framework of a moving window over the frames of data. The solution of these NP-hard,  
 20 noisy, large scale, and sparse problems to the noise level in the problem is fundamental to superior track estimation and identification. Thus, one must utilize the special structure in the problems as well take advantage of special information

that is available. Since these moving windows are overlapping, there are some algorithm efficiencies that can be identified and that take advantages of the overlap in the windows from one frame of reports to the next. Here is an  
 5 example of the use of a primal solution of one problem to warm start the solution of the next problem in the sequence.

Suppose we have solved problem (3.10) and have enumerated all those zero-one variables in the solution of (3.10) as in (3.11). Add the zero index  $l_{k+1}=0$ , so that the enumeration is

$$10 \quad \left\{ (l_k(l_{k+1}), i_{k+1}(l_{k+1}), \dots, i_{k+N}(l_{k+1})) \right\}_{l_{k+1}=0}^{L_{k+1}}. \quad (3.23)$$

With this enumeration one can define the cost by

$$c_{l_{k+1}i_{k+1+N}} = c_{l_k(l_{k+1})i_{k+1}(l_{k+1})\dots i_{k+N}(l_{k+1})i_{k+1+N}} \quad (3.24)$$

and the two-dimensional assignment problem

$$15 \quad \begin{aligned} \Phi_2 \equiv & \text{Minimize} \sum_{l_{k+1}=0}^{L_{k+1}} \sum_{i_{k+1+N}=0}^{M_{k+1+N}} c_{l_{k+1}i_{k+1+N}}^2 z_{l_{k+1}i_{k+1+N}}^2 \equiv V_2(z^2) \\ \text{Subject to} \quad & \sum_{i_{k+1+N}=0}^{M_{k+1+N}} z_{l_{k+1}i_{k+1+N}}^2 = 1, \quad l_{k+1} = 1, \dots, L_{k+1}, \\ & \sum_{l_{k+1}=0}^{L_{k+1}} z_{l_{k+1}i_{k+1+N}}^2 = 1, \quad i_{k+1+N} = 1, \dots, M_{k+1+N}, \\ & z_{l_{k+1}i_{k+1+N}}^2 \in \{0, 1\} \text{ for all } l_{k+1}, i_{k+1+N}. \end{aligned} \quad (3.25)$$

Let  $w$  be an optimal or feasible solution to this two-dimensional assignment problem and define



$$\hat{z}_{i_{k+1} \dots i_{k+N} i_{k+1+N}} = \begin{cases} 1, & \text{if } (i_{k+1}, \dots, i_{k+N}) = (i_{k+1}(l_{k+1}), \dots, i_{k+N}(l_{k+1})) \\ & \text{and } w_{l_{k+1} i_{k+1+N}} = 1 \text{ for some } l_{k+1} = 1, \dots, L_{k+1} \\ & \text{or if } (l_{k+1}, i_{k+1+N}) = (0, 0) \\ 0, & \text{otherwise.} \end{cases} \quad (3.26)$$

This need not satisfy the constraints in that there are usually many objects left unassigned. Thus, one can complete the assignment by using the zero-one variables in (3.10) with  $k$  replaced by  $k+1$  with exactly one nonzero index corresponding to any unassigned object or data report.

For the dual solutions, the multipliers arising from the solution of the two-dimensional assignment problem (3.25) corresponding to the second variable, i.e.,  $\left\{ u_{i_{k+1+N}}^{k+1+N} \right\}_{i_{k+1+N}=1}^{M_{k+1+N}}$ . These are good initial values to use in a relaxation scheme (A.B. Poore and N.Rijavec. Partitioning multiple data sets: multidimensional assignments and Lagrangian relaxation, in quadratic assignment and related problems; P.M. Pardalos and H. Wolkowicz, editors, *DIMACS series in Discrete Mathematics and Theoretical Computer Science*, 16:25-37, 1994; A.J. Robertson III. A class of lagrangian relaxation algorithms for the multidimensional assignment problem. *Ph.D. Thesis*, Colorado State University, Fr. Collins, CO, 1995). Finally, note that one can also develop a warm start for problem (3.20) in a similar fashion.

IV. REVIEW OF NEW RELAXATION SCHEMESIV.1. Introduction

Multidimensional assignment problems govern the central problem of data association in multisensor and multitarget tracking, i.e., the problem of partitioning observations from multiple scans of the surveillance region and from either single or multiple sensors into tracks and false alarms. This fundamental problem can be stated as (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994)

$$\text{Minimize} \quad \sum_{i_1=0}^{M_1} \cdots \sum_{i_N=0}^{M_N} c_{i_1 \cdots i_N} z_{i_1 \cdots i_N}$$

Subject to:

$$\sum_{i_2=0}^{M_2} \cdots \sum_{i_N=0}^{M_N} z_{i_1 \cdots i_N} = 1, i_1 = 1, \dots, M_1, \quad (4.1)$$

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{p-1}=0}^{M_{p-1}} \sum_{i_{p+1}=0}^{M_{p+1}} \cdots \sum_{i_N=0}^{M_N} z_{i_1 \cdots i_N} = 1$$

for  $i_p = 1, \dots, M_p$  and  $p = 2, \dots, N-1$ ,

$$\sum_{i_1=0}^{M_1} \cdots \sum_{i_{N-1}=0}^{M_{N-1}} z_{i_1 \cdots i_N} = 1, i_N = 1, \dots, M_N,$$

$z_{i_1 \cdots i_N} \in \{0, 1\}$  for all  $i_1, \dots, i_N$ ,

where  $c_{0 \cdots 0}$  is arbitrarily defined to be zero and is included for notational convenience. One can modify this formulation

5 to include *multi-assignments* of one, some, or all of the actual reports. The zero index is used in representing missing data, false alarms, initiating and terminating tracks as described in (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P.

10 Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and

15 multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994). In these problems, we assume

that all zero-one variables  $z_{i_1 \dots i_N}$  with precisely one nonzero index are free to be assigned and that the corresponding cost coefficients are well-defined. (This is a valid assumption in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).) Although not required, these cost coefficients with exactly one nonzero index can be translated to zero by cost shifting (A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising from multitarget tracking. *SIAM Journal of Optimization*, 3, No. 3:544-563, 1993) without changing the optimal assignment. Finally, this formulation is of sufficient generality to include the symmetric problem and the asymmetric inequality problem (A.J. Robertson III. A class of lagrangian algorithms for the multidimensional assignment problem. *Ph.D. Thesis*, Colorado State University, Fr. Collins, CO, 1995).

The data association problems in tracking that are formulated as Equation (4.1) have several characteristics. They are normally sparse, the cost coefficients are noisy and the problem is NP-hard (M.R. Galey and D.S. Johnson. *Computers and Intractability*. W.H. Freeman and Company, San Francisco, CA, 1979), but it must be solved in real-time. The only known methods for solving this NP-hard problem optimally are enumerative in nature, with branch-and-bound being the most efficient; however, such methods are much too slow for real-time applications. Thus one must resort to suboptimal approaches. Ideally, any such algorithm should solve the problem to within the noise level, assuming, of course, that one can measure this noise level in the physical problem and that the mathematical method provides a way to decide if the criterion has been met.

There are many algorithms that can be used to construct suboptimal solutions to NP-hard combinatorial optimization problems. These include greedy (and its many variants), relaxation, simulated annealing, tabu search, genetic algorithms, and neural network algorithms (C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithm and Complexity*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982;

J.Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, Reading, MA, 1984; C.R. Reeves ed. *Modern Heuristic Techniques for Combinatorial Problems*. Halstead Press, Wiley, NY, 1993). For the three-

5 dimensional assignment problem, Pierskalla (W. Pierskalla. The tri-substitution method for three-dimensional assignment problem. *Journal du CORS*, 5:71-81, 1967) developed the tri-substitution method, which is a variant of the simplex method. Frieze and Yadegar (A.M. Frieze and J. Yadegar. An algorithm

10 for solving 3-dimensional assignment problems with application to scheduling a teaching practice. *Journal of the Operational Research Society*, 39:989-955, 1981) introduced a method based on Lagrangian relaxation in which a feasible solution is recovered using information provided by the relaxed solution.

15 Our choice of approaches is strongly influenced by the need to balance real-time performance and solution quality. Lagrangian relaxation based methods have been used successfully in prior tracking applications (K.R. Pattipati, S. Deb, and Y. Bar-Shalom. A s-dimensional assignment

20 algorithm for track initiation. In *Proceedings of the IEEE Systems Conference, Kobe, Japan*, pages 127-130, 1992; Y. Bar-Shalom, S. Deb, K.R. Pattipati, and H. Tsanakakis. A new algorithm for the generalized multidimensional assignment

problem. In *Proceedings of the IEEE International Conference on Systems, Math, and Cybernetics, Chicago*, pages 132-136, 1992; A.B. Poore and N. Rijavec. A numerical study of some data association problems arising in multitarget tracking.

5 *Large Scale Optimization: State of the Art*, W.W. Hager, D.W. Hearn and P.M. Pardalos, editors. Kluwer Academic Publishers B.V., Boston, pages 339-361, 1994; A.B. Poore and N. Rijavec. Partitioning multiple data sets: multidimensional assignments and lagrangian relaxation. In P.M. Pardalos and H. Wolkowicz,

10 editors, *Quadratic assignment and related problems: DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 16, pages 25-37, 1994; A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems from multitarget

15 tracking. *SIAM Journal of Optimization*, 3, No. 3:544-563, 1993). An advantage of these methods is that they provide both an upper and lower bound on the optimal solution, which can then be used to measure the solution quality. These works extend the method of Frieze and Yadegar (A.M. Frieze and

20 J. Yadegar. An algorithm for solving 3-dimensional assignment problems with application to scheduling a teaching practice. *Journal of the Operational Research Society*, 32:989-995, 1981) to the multidimensional case.

IV.2. Probabilistic Framework for Data Association.

The goal of this section is to explain the formulation of the data association problems that governs large classes of data association problems in centralized or hybrid centralized-sensor level multisensor/multitarget tracking. The presentation is brief; technical details are presented for both track initiation and maintenance in the work of (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994). The formulation is of sufficient generality to cover the MHT work of Reid, Blackman and Stein (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986) and modifications by Kurien (T.Kurien. Issues in the designing of practical multitarget tracking algorithms. In *Multitarget-Multisensor Tracking: Advanced Applications* by Y. Bar-Shalom. Artech House, MA, 1990) to



include maneuvering targets. As suggested by Blackman (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986), this formulation can also be modified to include target features (e.g., size and type) into the scoring function. The recent work (A.B. Poore and O.E. Drummond. Track initiation and maintenance using multidimensional assignment problems. In D.W. Hearn, W.W. Hager, and P.M. Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.) significantly extends the work of this section to new approaches for multiple sensor centralized tracking. Future work will involve extensions to track-to-track correlation.

The data association problems for multisensor and multitarget tracking considered in this work are generally posed as that of maximizing the posterior probability of the surveillance region (given the data) according to

$$\text{Maximize } \left\{ P(\Gamma=\gamma | Z^N) \mid \gamma \in \Gamma^* \right\}, \quad (4.2)$$

where  $Z^N$  represents  $N$  data sets,  $\gamma$  is a partition of indices of the data (and thus induces a partition of the data),  $\Gamma^*$  is the finite collection of all such partitions,  $\Gamma$  is a discrete

random element defined on  $\Gamma^*$ , and  $P(\Gamma = \gamma | Z^N)$  is the posterior probability of a partition  $\gamma$  being true given the data  $Z^N$ . The term partition is defined below; however, this framework is currently sufficiently general to cover set packings and coverings (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995).

Consider  $N$  data sets  $Z(k)$  ( $k=1, \dots, N$ ) each of  $M_k$  reports  $\left\{ z_{i_k}^k \right\}_{i_k=1}^{M_k}$ , and let  $Z^N$  denote the cumulative data set defined by

$$Z(k) = \left\{ z_{i_k}^k \right\}_{i_k=1}^{M_k} \text{ and } Z^N = \{ Z(1), \dots, Z(N) \}, \quad (4.3)$$

respectively. In multisensor data fusion and multitarget tracking the data sets  $Z(k)$  may represent different classes of objects, and each data set can arise from different sensors. For track initiation the objects are measurements that must be partitioned into tracks and false alarms. In our formulation

of track maintenance (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994;

5 A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995), which uses a moving

10 window over time, one data set will be tracks and remaining data sets will be measurements which are assigned to existing tracks, as false measurements, or to initiating tracks. In sensor level tracking, the objects to be fused are tracks (S.S. Blackman. *Multiple Target Tracking with Radar*

15 *Applications*. Artech House, Norwood, MA, 1986). In centralized fusion (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986), the objects may all be measurements that represent targets or false reports, and the problem is to determine which

20 measurements emanate from a common source.

We specialize the problem to the case of set partitioning (A.B. Poore. Multidimensional assignment formulation of data

association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995) defined in the following way. First, for notational convenience in representing tracks, we add a zero index to each of the index sets in Equation (4.3), a dummy report  $z_0^k$  to each of the data sets  $Z(k)$  in Equation (4.3), and define a "track of data" as  $\left( z_{i_1}^1, \dots, z_{i_N}^N \right)$ , where  $i_k$  can now assume the value of 0. A partition of the data will refer to a collection of tracks of data wherein each report occurs exactly once in one of the tracks of data and such that all data is used up; the occurrence of dummy report is unrestricted. The dummy report  $z_0^k$  serves several purposes in the representation of missing data, false reports, initiating tracks, and terminating tracks (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget

tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995). The reference partition  
 5 is that in which all reports are declared to be false.

Next under appropriate independence assumptions one can show (A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking  
 10 and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*,  
 15 *American Mathematical Society, Providence, R.I.*, 19:169-198, 1995)

$$\frac{P(\Gamma=\gamma|Z^N)}{P(\Gamma=\gamma^0|Z^N)} \equiv L_\gamma \equiv \prod_{(i_1 \dots i_N) \in \gamma} L_{i_1 \dots i_N}, \quad (4.4)$$

where  $\gamma^0$  is a reference partition,  $L_{i_1 \dots i_N}$  is a likelihood ratio  
 20 containing probabilities for detection, maneuvers, and termination as well as probability density functions for measurement errors, track initiation and termination. Then if  
 $c_{i_1 \dots i_N} = -\ln L_{i_1 \dots i_N}$ ,

$$-\ln \left[ \frac{P(\Gamma = \gamma | Z^N)}{P(\Gamma = \gamma^0 | Z^N)} \right] = \sum_{(i_1, \dots, i_N) \in \gamma} c_{i_1 \dots i_N}. \quad (4.5)$$

5 Using Equation (4.4) and the zero-one variable  $z_{i_1 \dots i_N} = 1$  if  $(i_1, \dots, i_N) \in \gamma$  and 0 otherwise, one can then write the problem (4.4) as the following  $N$ -dimensional assignment problem:

$$\begin{aligned} & \text{Minimize} && \sum_{i_1 \dots i_N} c_{i_1 \dots i_N} z_{i_1 \dots i_N} \\ & \text{Subject To} && \sum_{i_2 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_1 = 1, \dots, M_1), \\ & && \sum_{i_1 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_2 = 1, \dots, M_2), \\ & && \sum_{i_1 \dots i_{p-1} i_{p+1} \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_p = 1, \dots, M_p \text{ and } p = 2, \dots, N-1), \\ & && \sum_{i_1 i_2 \dots i_{N-1}} z_{i_1 \dots i_N} = 1 \quad (i_N = 1, \dots, M_N), \\ & && z_{i_1 \dots i_N} \in \{0, 1\} \text{ for all } i_1, \dots, i_N, \end{aligned} \quad (4.6)$$

10 where  $c_{0 \dots 0}$  is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each-non-dummy report occurs exactly once in a "track of data." One can modify this formulation to include *multi-assignments* of one, some, or all the actual reports. The assignment problem  
15 Equation (4.6) is changed accordingly. For example, if  $z_{i_k}^k$  is

to be assigned no more than, exactly, or no less than  $n_{i_k}^k$  times, then the "=1" in the constraint (4.6) is changed to  $\leq, =, \geq n_{i_k}^k$ , " respectively. Modifications for group tracking and multi-resolution features of the surveillance region will  
 5 be addressed in future work. In making these changes, one must pay careful attention to the independence assumptions that need not be valid in many applications.

Expressions for the likelihood ratios  $L_{i_1, \dots, i_N}$  can be found in the work of Poore (A.B. Poore. Multidimensional assignment  
 10 formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B.  
 15 Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995). These expressions include the developments of Reid, Kurien (T. Kurien. Issues in the designing of practical multitarget tracking algorithms. In  
 20 *Multitarget-Multisensor Tracking: Advanced Applications* by Y. Bar-Shalom. Artech House, MA, 1990), and Stein and Blackman (S.S. Blackman. *Multiple Target Tracking with Radar Applications*. Artech House, Norwood, MA, 1986). What's more,

they are easily modified to include target features and to account for different sensor types. In *track initiation*, the  $N$  data sets all represent reports from  $N$  sensors, possibly all the same. For *track maintenance*, we use a *sliding window* of

5  $N$  data sets and one data set containing established tracks.

(A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994; A.B. Poore. Multidimensional

10 assignments and multitarget tracking, partitioning data sets, I.J. cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995). The formulation is the same as above except that the

15 dimension of the assignment problem is now  $N+1$ .

#### IV.3. Overview of the Lagrangian Relaxation Algorithm.

Having discussed the  $N$ -dimensional assignment problem (3.1), we now turn to a description of the Lagrangian

20 relaxation algorithm. The algorithm will proceed iteratively for a loop  $k=1, \dots, N-2$ . At the completion, there remains one two-dimensional assignment problem that provides the last



step which yields an optimal (sometimes) or near-optimal solution to the original  $N$ -dimensional assignment problem. In step  $k$  of this loop (summarized in Section IV.3) one starts with the following  $(N-k+1)$ -dimensional assignment problem with one change in notation. If  $k=1$ , then the index notation  $l_1$  and  $L_1$  are to be replaced by  $i_1$  and  $M_1$ , respectively.

$$\begin{aligned}
 &\text{Minimize} && \sum_{l_k i_{k+1} \dots i_N} c_{l_k i_{k+1} \dots i_N}^{N-k+1} z_{l_k i_{k+1} \dots i_N}^{N-k+1} \\
 &\text{Subject To} && \sum_{i_{k+1} \dots i_N} z_{l_k i_{k+1} \dots i_N}^{N-k+1} = 1, \quad l_k = 1, \dots, L_k, \\
 &&& \sum_{l_k i_{k+2} \dots i_N} z_{l_k i_{k+2} \dots i_N}^{N-k+1} = 1, \quad i_{k+1} = 1, \dots, M_{k+1}, \\
 &&& \sum_{l_k i_{k+1} \dots i_{p-1} i_{p+1} \dots i_N} z_{l_k i_{k+1} \dots i_N}^{N-k+1} = 1, \\
 &&& \text{for } i_p = 1, \dots, M_p \text{ and } p = k+2, \dots, N-1, \\
 &&& \sum_{l_k i_{k+1} \dots i_{N-k+1}} z_{l_k i_{k+1} \dots i_N}^{N-k+1} = 1, \quad i_N = 1, \dots, M_N, \\
 &&& z_{l_k i_{k+1} \dots i_N} \in \{0, 1\} \text{ for all } l_k, i_{k+1}, \dots, i_N.
 \end{aligned} \tag{4.7}$$

To ensure that a feasible solution of Equation (4.7) always exists for a sparse problem, all variables with exactly one nonzero index (i.e., variables of the form  $z_{l_k 0 \dots 0}^{N-k+1}$  for  $l_k = 1, \dots, L_k$  and  $z_{0 \dots 0 i_p 0 \dots 0}^{N-k+1}$  for  $i_p = 1, \dots, M_{i_p}$  and  $p = k+1, \dots, N$ ) are assumed free to be assigned with the corresponding cost coefficients being well-defined. This assumption is valid in

the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*,  
 5 *American Mathematical Society, Providence, R.I.*, 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).

10       Section IV.3.1 presents some of the properties associated with the Lagrangian relaxation of (4.7) based on relaxing the last  $(N-k)$ -sets of constraints to a two-dimensional one. Section IV.3.2 describes a new approach to the problem of recovering a high quality feasible solution of the original  
 15  $(N-k+1)$ -dimensional problem given a feasible solution (optimal or suboptimal) of the relaxed two-dimensional problem is described hereinafter. A summary of the relaxation algorithm is given in Section IV.3.3, and in Section IV.3.4, we establish the maximization of the Lagrangian dual (an  
 20 important aspect of the relaxation procedure) to be an unconstrained nonsmooth optimization problem and then present a method for computing the subgradients.

IV.3.1 The Lagrangian Relaxed Assignment Problem

The  $(N-k+1)$ -dimensional problem (4.7) has  $(N-k+1)$  sets of constraints. A  $(M_p+1)$ -dimensional multiplier vector (i.e.,  $u^p \in \mathbb{R}^{M_p+1}$ ) associated with the  $p^{th}$  constraint set will be  
 5 denoted by  $u^p = (u_0^p, u_1^p, \dots, u_{M_p}^p)^T$  with  $u_0^p \equiv 0$  being fixed for each  $p = k + 2, \dots, N$  and included for notational convenience only. Now, the  $(N-k+1)$ -dimensional assignment problem (4.7) is relaxed to a two-dimensional assignment problem by incorporating  $(N-k-1)$  sets of constraints into the objective  
 10 function via the Lagrangian. Although any  $(N-k-1)$  constraint sets can be relaxed, we choose the *last*  $(N-k-1)$  sets of constraints for convenience. The *relaxed problem* is

$$\begin{aligned}
\Phi_{N-k+1}(u^{k+2}, \dots, u^N) &\equiv \text{Minimize } \phi_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) \equiv \\
&\text{Minimize } \sum_{l_k i_{k+1} \dots i_N} c_{l_k i_{k+1} \dots i_N}^{N-k+1} z_{l_k i_{k+1} \dots i_N}^{N-k+1} \\
&\quad + \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p \left[ \sum_{l_k i_{k+1} \dots i_{p-1} i_{p+1} \dots i_N} z_{l_k i_{k+1} \dots i_N}^{N-k+1} - 1 \right] \equiv \quad (4.8) \\
&\text{Minimize } \sum_{l_k i_{k+1} \dots i_N} \left[ c_{l_k i_{k+1} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p \right] z_{l_k i_{k+1} \dots i_N}^{N-k+1} \\
&\quad - \sum_{p=k-2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p \\
&\text{Subject To } \sum_{i_{k+1} i_{k+2} \dots i_N} z_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} = 1, \quad l_k = 1, \dots, L_k, \\
&\quad \sum_{l_k i_{k+2} \dots i_N} z_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} = 1, \quad i_{k+1} = 1, \dots, M_{k+1}.
\end{aligned}$$

One of the major steps in the algorithm is the maximization of  $\Phi_{N-k+1}(u^{k+2}, \dots, u^N)$  with respect to the multipliers  $(u^{k+2}, \dots, u^N)$ . It turns out that  $\Phi_{N-k+1}$  is a concave, continuous, and piecewise affine function of the multipliers  $(u^{k+2}, \dots, u^N)$ , so that the maximization of  $\Phi_{N-k+1}$  is a problem of nonsmooth optimization. Since many of these algorithms require a function value and a subgradient of  $\Phi_{N-k+1}$  at any required multiplier value  $(u^{k+2}, \dots, u^N)$ , we address this problem in the next subsection. We note, however, that

there are other ways to maximize  $\Phi_{N-k+1}$  and the next subsection just addresses one such method.

#### IV.3.2 Properties of the Lagrangian Relaxed Assignment Problem

5 For a function evaluation of  $\Phi_{N-k+1}$ , we show that an optimal (or suboptimal) solution of this relaxed problem (4.8) can be constructed from that of a two-dimensional assignment problem. Then, the nonsmooth characteristics of  $\Phi_{N-k+1}$  are addressed, followed by a method for computing the function  
10 value and a subgradient.

**Evaluation of  $\Phi_{N-k+1}$ .** Define for each  $(l_k, i_{k+1})$  an index  $(j_{k+2}, \dots, j_N) = (j_{k+2}(l_k, i_{k+1}), \dots, j_N(l_k, i_{k+1}))$  and a new cost function  $c_{l_k i_{k+1}}^2$  by

$$(j_{k+2}(l_k, i_{k+1}), \dots, j_N(l_k, i_{k+1})) = \underset{\substack{C_{l_k i_{k+1} j_{k+2} \dots j_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \\ \text{and } p=k+2, \dots, N}}{\operatorname{argmin}} \left\{ u_{i_p}^p \mid i_p = 0, 1, \dots, M_p \right\}, \quad (4.9)$$

$$\begin{aligned} c_{l_k i_{k+1}}^2 &= c_{l_k i_{k+1} j_{k+2}(l_k, i_{k+1}) \dots j_N(l_k, i_{k+1})}^{N-k+1} \\ &\quad + \sum_{p=k+2}^N u_{j_p}^p(l_k, i_{k+1}) \text{ for } (l_k, i_{k+1}) \neq (0, 0), \\ c_{00}^2 &= \sum_{i_{k+1} \dots i_N} \operatorname{Minimum} \left\{ 0, c_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \right\}, \end{aligned}$$

Given an index pair  $(l_k, i_{k+1}), (j_{k+2}, \dots, j_N)$  need not be unique, resulting in the potential generation of several subgradients (4.17). Then,

$$\begin{aligned}
 \hat{\phi}_{N-k+1}(u^{k+2}, \dots, u^N) = \\
 \text{Minimize } \hat{\phi}_{N-k+1}(z^2; u^{k+2}, \dots, u^N) \\
 \equiv \sum_{l_k=0}^{L_k} \sum_{i_{k+1}=0}^{M_{k+1}} C_{l_k i_{k+1}}^2 z_{l_k i_{k+1}}^2 - \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p \quad (4.10) \\
 \text{Subject To } \sum_{i_{k+1}=0}^{M_{k+1}} z_{l_k i_{k+1}}^2 = 1, l_k = 1, \dots, L_k, \\
 \sum_{l_k=0}^{L_k} z_{l_k i_{k+1}}^2 = 1, i_{k+1} = 1, \dots, M_{k+1}, \\
 z_{l_k i_{k+1}}^2 \in \{0, 1\} \text{ for all } l_k, i_{k+1}.
 \end{aligned}$$

As an aside, two observations are in order. The first is that the search procedure needed for the computation of the relaxed cost coefficients in (4.9) is the most computationally intensive part of the entire relaxation algorithm. The second is that a feasible solution  $z^{N-k+1}$  of a sparse problem (4.7) yields a feasible solution  $z^2$  of (4.10) via the construction

$$z_{l_k i_{k+1}}^2 = \begin{cases} 1, & \text{if } z_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} = 1 \text{ for some } (i_{k+2}, \dots, i_N), \\ 0, & \text{otherwise.} \end{cases}$$

thus, there are generally solutions other than the one nonzero index solution.

The following Theorem 4.1 gives a method for evaluating  $\phi_{N-k+1}$  and states that an optimal solution of (4.8) can be computed from that of (4.10). Furthermore, if the solution of either of these two problems is  $\epsilon$ -optimal, then so is the other. The  
 5 converse is contained in Theorem 4.2.

**Theorem 4.1.** Let  $w^2$  be a feasible solution to the two-dimensional assignment problem (4.10) and define  $w^{N-k+1}$  by

$$\begin{aligned}
 w_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} &= w_{l_k i_{k+1}}^2 \quad \text{if } (i_{k+2}, \dots, i_N) = (j_{k+2}, \dots, j_N) \\
 &\quad \text{and } (l_k, i_{k+1}) \neq (0, 0); \\
 w_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} &= 0 \quad \text{if } (i_{k+2}, \dots, i_N) \neq (j_{k+2}, \dots, j_N) \\
 &\quad \text{and } (l_k, i_{k+1}) \neq (0, 0); \\
 w_{00 i_{k+2} \dots i_N}^{N-k+1} &= 1 \quad \text{if } c_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \leq 0, \\
 w_{00 i_{k+2} \dots i_N}^{N-k+1} &= 0 \quad \text{if } c_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p > 0.
 \end{aligned} \tag{4.11}$$

Then  $w^{N-k+1}$  is a feasible solution of the Lagrangian relaxed problem and

$$10 \quad \phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) = \phi_{-k-1}(w^2; u^{k+2}, \dots, u^N).$$

If, in addition,  $w^2$  is optimal for the two-dimensional problem, then  $w^{N-k+1}$  is an optimal solution of the relaxed problem and

$$\phi_{N-k+1}(u^{k+2}, \dots, u^N) = \phi_{-k+1}(u^{k+2}, \dots, u^N).$$

**Proof.** Let  $w^{N-k+1}$  and  $w^2$  be as in the hypotheses, and let

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) \text{ and}$$

$$\phi_{N-k+1}(z^2; u^{k+2}, \dots, u^N)$$

denote the objective function values of (4.8) and (4.10),  
 5 respectively. Direct verification shows that  $w^{N-k+1}$  satisfies  
 the constraints in (4.8) and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}; \dots, u^N) = \phi_{N-k+1}(w^2; u^{k+2}, \dots, u^N).$$

For the remainder of the proof, assume that  $w^2$  is optimal for  
 (4.10). Let  $x^{N-k+1}$  satisfy the constraints in (4.8) and define

10

$$x_{l_k i_{k+1}}^2 = \sum_{i_{k+2} \dots i_N} x_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} \text{ for } (l_k, i_{k+1}) \neq (0, 0),$$

$$x_{00}^2 = 1 \text{ if } (l_k, i_{k+1}) = (0, 0) \text{ and}$$

$$c_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \leq 0 \text{ for some } (i_{k+2}, \dots, i_N), \quad (4.12)$$

$$x_{00}^2 = 0 \text{ if } (l_k, i_{k+1}) = (0, 0) \text{ and}$$

15

$$c_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p > 0 \text{ for all } (i_{k+2}, \dots, i_N).$$

Note that  $x^2$  satisfies the constraints in (4.10) and

20

$$\begin{aligned} \phi_{N-k+1}(x^{N-k+1}; u^{k+2}, \dots, u^N) &\geq \phi_{N-k+1}(x^2; u^{k+2}, \dots, u^N) \\ &\geq \phi_{N-k+1}(w^2; u^{k+2}, \dots, u^N) \\ &= \phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) \end{aligned}$$



for any feasible solution  $x^{N-k+1}$  of the constraints (4.8). This implies  $w^{N-k+1}$  is an optimal solution of (4.8). Next,

$$\phi_{N-k+1}(u^{k+2}, \dots, u^N) = \phi'_{N-k+1}(u^{k+2}, \dots, u^N)$$

follows immediately from

$$5 \quad \phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) = \phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N)$$

for an optimal solution  $w^2$  of (4.10). With the exception of one equality being converted to an inequality, the following theorem is a converse of Theorem 4.1.

**Theorem 4.2.** Let  $w^{N-k+1}$  be a feasible solution to problem (4.8)

10 and define  $w^2$  by

$$w_{l_k i_{k+1}}^2 = \sum_{i_{k+2} \dots i_N} w_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} \text{ for } (l_k, i_{k+1}) \neq (0, 0),$$

$$w_{00}^2 = 1 \text{ if } (l_k, i_{k+1}) = (0, 0) \text{ and}$$

$$C_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \leq 0 \text{ for some } (i_{k+2}, \dots, i_N), \quad (4.13)$$

15

$$w_{00}^2 = 0 \text{ if } (l_k, i_{k+1}) = (0, 0) \text{ and}$$

$$C_{00 i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p > 0 \text{ for all } (i_{k+2}, \dots, i_N).$$

Then  $w^2$  a  
feasible

20 solution of the problem (4.10) and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) \geq \phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N).$$

If, in addition,  $w^{N-k+1}$  is optimal for (4.8), then  $w^2$  is an optimal solution of (4.10),

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) = \phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N) \text{ and}$$

$$\phi_{N-k+1}(u^{k+2}, \dots, u^N) = \phi'_{N-k+1}(u^{k+2}, \dots, u^N).$$

**Proof:** Let  $w^{N-k+1}$  and  $w^2$  be as in the hypotheses, and let  $\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N)$  and  $\phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N)$  denote the  
 5 objective function values of (4.8) and (4.10), respectively. Direct verification shows that  $w^2$  satisfies the constraints in (4.10) and

$$\phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N) \geq \phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N).$$

For the remainder of the proof, assume that  $w^{N-k+1}$  is optimal  
 10 for (4.8) and construct  $w^2$  as above. Using Theorem 4.1, construct  $\bar{w}^{N-k+1}$  by replacing  $w^{N-k+1}$  in (4.11) with  $\bar{w}^{N-k+1}$ . Then, from that theorem

$$\begin{aligned} \phi_{N-k+1}(\bar{w}^{N-k+1}; u^{k+2}, \dots, u^N) &= \phi'_{N-k+1}(w^2; u^{k+2}, \dots, u^N) \\ &\leq \phi_{N-k+1}(w^{N-k+1}; u^{k+2}, \dots, u^N). \end{aligned}$$

15 Optimality of  $w^{N-k+1}$  then implies the last inequality is in fact an equality. This proves the theorem.

**The Nonsmooth Optimization Problem.** Next, we address the nonsmooth properties of the function  $\phi_{N-k+1}(u^{k+2}, \dots, u^N)$  as explained in the following theorem.

20 **Theorem 4.3.** (G.L. Nemhauser and L.A. Wolsey. Integer and Combinatorial Optimization, Section II.3.6. Wiley, New York, NY, 1988). Let  $\phi_{N-k+1}$  be as defined in (4.7), let  $V_{N-k+1}(z^{N-k+1})$

be the objective function value of the  $(N-k+1)$ -dimensional assignment problem in equation (4.7) corresponding to a feasible solution  $z^{N-k+1}$  of the constraints in (4.7), and let  $\bar{z}^{N-k+1}$  be an optimal solution of (4.7). Then,  $\Phi_{N-k+1}(u^{k+2}, \dots, u^N)$  is piecewise affine, concave and continuous in  $\{u^{k+2}, \dots, u^N\}$  and

$$\Phi_{N-k+1}(u^{k+2}, \dots, u^N) \leq V_{N-k+1}(\bar{z}^{N-k+1}) \leq V_{N-k+1}(z^{N-k+1}). \quad (4.14)$$

Most of the algorithms for nonsmooth optimization are based on generalized gradients, called *subgradients*, given by the following definition for a concave function.

**Definition 4.1.** At  $\square = (u^{k+2}, \dots, u^N)$  the set  $\partial\Phi_{N-k+1}(u)$  is called a subdifferential of  $\Phi_{N-k+1}$  and is defined for the concave function  $\Phi_{N-k+1}(u)$  by

$$\partial\Phi_{N-k+1}(u) = \{g \in \mathbb{R}^{M_{k+2}+1} \times \dots \times \mathbb{R}^{M_N+1} \mid \Phi_{N-k+1}(w) - \Phi_{N-k+1}(u) \leq g^T(w-u) \text{ for all } w \in \mathbb{R}^{M_{k+2}+1} \times \dots \times \mathbb{R}^{M_N+1}\}, \quad (4.15)$$

where  $g_0^P = w_0^P = u_0^P = 0$  are all permanently fixed. (Recall that these were used for notational convenience only.) A vector  $g \in \partial\Phi_{N-k+1}$  is called a subgradient.

There is a large literature on such problems, e.g.,  
 (J.-B. Hiriart-Urruty and C. Lemarechal. *Concex Analysis and  
 Minimization Algorithms I& II*. Springer-Verlag, Berlin, 1993;  
 K.C. Kiwiel. Methods of descent for nondifferentiable  
 5 optimization. In *Lecture Notes in Mathematics 1133*, A. Dold  
 and B. Eckmann, eds. Springer-Verlag, Berlin, 1985;  
 C.Lemarechal and R. Mifflin. *Nonsmooth Optimization*. Pergamon  
 Press, Oxford, UK, 1978; B.T. Polyak. Subgradient method:  
 A survey of Soviet research. In C. Lemarechal and R. Mifflin,  
 10 editors, *Nonsmooth Optimization*, pages 5-29, N.Y., 1978.  
 Pergamon Press.; H.Schramm and J. Zowe. A version of the  
 bundle idea for minimizing a nonsmooth function: Conceptual  
 idea, convergence analysis, numerical results. *SIAM Journal on  
 Optimization*, 2, No.1:121-152, February, 1992; N.Z. Shor.  
 15 *Minimization Methods for Non-Differentiable Functions*.  
 Springer-Verlag, Nem York, 1985; P.Wolfe. A method of  
 conjugate subgradients for minimizing nondifferentiable  
 functions. *Mathematical Programming Study*, 3:145-173, 1975),  
 and we have tried a variety of methods including subgradient  
 20 methods (N.Z. Shor. *Minimization Methods for Non-  
 Differentiable Functions*. Springer-Verlag, New York, 1985)  
 and bundle methods (J.-B. Hiriart-Urruty and C. Lemarechal.  
*Convex Analysis and Minimization Algorithms I& II*. Springer-

Verlag, Berlin, 1993; K.C. Kiwiel. Methods of descent for nondifferentiable optimization. In *Lecture Notes in Mathematics 1133*, A. Dold and B. Eckmann, eds. Springer-Verlag, Berlin, 1985). Of these, we have determined that for a fixed  
 5 number of nonsmooth iterations (e.g., twenty), the bundle trust method of Schramm and Zowe (H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2, No. 1:121-152,  
 10 February, 1992) provides excellent quality solutions with the fewest number of function and subgradient evaluations, and is therefore our currently recommended method.

#### **An Algorithm for Computing $\phi_{N-k+1}$ and a Subgradient.**

Most current software for maximizing the concave function  
 15  $\phi_{N-k+1}$  requires the value of the function and a subgradient at a point  $(u^{k+2}, \dots, u^N)$ . Based on the previous two sections, this can be summarized as follows.

1. Starting with problem (4.7), form the relaxed problem (4.8);
- 20 2. To solve (4.8) optimally, defined the two-dimensional assignment problem (4.10) via the transformation (4.9);
3. Solve the two-dimensional problem (4.10) optimally;

4. Reconstruct the optimal solution, say  $\hat{w}^{N-k+1}$ , of (4.8) via equation (4.9) as in Theorem 4.1;

5. Compute the function

$$\begin{aligned}
 5 \quad \phi_{N-k+1}(u^{k+2}, \dots, u^N) &\equiv \phi_{N-k+1}(\hat{w}^{N-k+1}; u^{k+2}, \dots, u^N) \\
 &\equiv \sum_{l_k^{i_{k+1}}, \dots, i_N} c_{l_k^{i_{k+1}}, \dots, i_N}^{N-k+1} \hat{w}_{l_k^{i_{k+1}}, \dots, i_N}^{N-k+1} \\
 &\quad + \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p \left[ \sum_{l_k^{i_{k+1}}, \dots, i_{p-1}^{i_{p+1}}, \dots, i_N} \hat{w}_{l_k^{i_{k+1}}, \dots, i_N}^{N-k+1} - 1 \right];
 \end{aligned} \tag{4.16}$$

10 6. A subgradient is given by

$$\begin{aligned}
 g_{i_p}^p &= \frac{\partial \phi_{N-k+1}(u^{k+2}, \dots, u^N)}{\partial u_{i_p}^p} = \left[ \sum_{l_k^{i_{k+1}}, \dots, i_{p-1}^{i_{p+1}}, \dots, i_N} \hat{w}_{l_k^{i_{k+1}}, \dots, i_N}^{N-k+1} - 1 \right] \\
 &\quad \text{for } i_p = 1, \dots, M_p \text{ and } p = k+2, \dots, N.
 \end{aligned} \tag{4.17}$$

Several remarks are in order. First, the optimal solution of the minimization problem (4.8) is required before one can remove the minimum sign, replace  $z^{N-k+1}$  by the minimizer  $\hat{w}^{N-k+1}$  and differentiate with respect to  $u_{i_p}^p$  to obtain a subgradient as in (4.17). If  $\hat{w}^{N-k+1}$  is an approximate solution of (4.8), then the subgradient and function values are only approximate with accuracy depending on that of  $\hat{w}^{N-k+1}$ . Although one can

evaluate the sums in (4.16) and (4.17) in a straight forward manner, another method is based on the following observation. Given the multiplier vector  $(u_{k+2}, \dots, u_N)$ , let  $\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\}_{l_{k+1}=0}^{L_{k+1}}$  be an enumeration of indices  $\{l_k, i_{k+1}\}$  of  $w^2$  (or the first 2 indices of  $w^{N-k+1}$  constructed in equation (4.9)) such that  $w_{l_k(l_{k+1}), i_{k+1}(l_{k+1})}^2 = 1$  for  $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) \neq (0, 0)$  and  $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) = (0, 0)$  for  $l_{k+1} = 0$  regardless of whether  $w_{00}^2 = 1$  or not. (The latter can only improve the recovered feasible solution.) The evaluation of the bracketed quantity in (4.17) for a specific  $i_p \geq 1$  and  $p = k+2, \dots, N$  is one minus the number of times the index value  $i_p$  appears in the  $(p-k+1)^{th}$  position of the  $(N-k+1)$ -tuple in the list  $\{l_k(l_{k+1}), i_{k+1}(l_{k+1}), j_{k+2}, \dots, j_N\}_{l_{k+1}=0}^{L_{k+1}}$ .

Finally, we have presented a method for computing one subgradient. If  $\hat{w}^{N-k+1}$  is the unique optimal solution of (4.8), then  $\Phi_{N-k+1}(u)$  is differentiable at  $u$ ,  $g$  is just the gradient at  $u$ , and the subdifferential  $\partial\Phi_{N-k+1}(u) = \{g\}$  is a singleton. If, corresponding to  $u$ ,  $\hat{w}^{N-k+1}$  is not unique, then there are finitely many such solutions, say  $\{\hat{w}^{N-k+1}(1), \dots, \hat{w}^{N-k+1}(m)\}$  with respective subgradients  $\{g(1), \dots, g(m)\}$ , the subdifferential  $\partial\Phi(u)$  is the convex hull of  $\{g(1), \dots, g(m)\}$  (J.L. Goffin. On convergence rates of subgradient optimization methods, mathematical programming. *Mathematical Programming*, 13:329-

347, 1977). These nonunique solutions arise in two ways. First, if the optimal solution of the two-dimensional assignment problem is not unique, then one can generate all optimal solutions of the two-dimensional assignment problem  
 5 (4.10). Corresponding to each of these solutions and to each index pair  $(l_k, i_{k+1})$  in each solution, compute the indices  $(j_{k+2}, \dots, j_N)$  in (4.9). If these  $j$ 's are not unique, then we can enumerate all the possible optimal solutions  $\hat{w}^{N-k+1}$  of (4.8). Given these solutions, one can then compute the  
 10 corresponding subgradients from (4.17). In most nonsmooth optimization algorithms, one uses an  $\epsilon$ -subdifferential.

**Definition 4.2.** At  $u = (u^{k+2}, \dots, u^N)$  the set  $\partial_\epsilon \Phi_{N-k+1}(u)$  is called an  $\epsilon$ -subdifferential of  $\Phi_{N-k+1}$  and is defined for the  
 15 concave function  $\Phi_{N-k+1}(u)$  by

$$\partial_\epsilon \Phi_{N-k+1}(u) = \{g \in \mathbb{R}^{M_{k+2}+1} \times \dots \times \mathbb{R}^{M_N+1} \mid \Phi_{N-k+1}(w) - \Phi_{N-k+1}(u) \leq g^T(w-u) + \epsilon \text{ for all } w \in \mathbb{R}^{M_{k+2}+1} \times \dots \times \mathbb{R}^{M_N+1}\},$$

where  $g_0^P = w_0^P = u_0^P = 0$  are all permanently fixed. (Recall that these were used for notational convenience only.) A vector  $g \in \partial_\epsilon \Phi_{N-k+1}(u)$  is called an  $\epsilon$ -subgradient.



IV.3.3 The Recovery Procedure.

The next objective is to explain a recovery procedure, i.e., given a feasible (optimal or suboptimal) solution  $w^2$  of (4.10) (or  $w^{N-k+1}$  of (4.8) constructed via Theorem 4.1), generate a feasible solution  $z^{N-k+1}$  of equation (4.7) which is close to  $w^2$  in a sense to be specified. We first assume that no variables in (4.7) are preassigned to zero; this assumption will be removed shortly. The difficulty with the solution  $w^{N-k+1}$  in Theorem 4.1 is that it need not satisfy the last  $(N-k-1)$  sets of constraints in (4.7). (Note, however, that if  $w^2$  is an optimal solution (4.10) and  $w^{N-k+1}$ , as constructed in Theorem 4.1, satisfies the relaxed constraints, then  $w^{N-k+1}$  is optimal for (4.7).) The recovery procedure described here is designed to preserve the 0-1 character of the solution  $w^2$  of (4.10) as far as possible: If  $w_{l_k i_{k+1}}^2 = 1$  and  $l_k \neq 0$  or  $i_{k+1} \neq 0$ , the corresponding feasible solution  $z^{N-k+1}$  of (4.7) is constructed so that  $z_{l_k i_{k+1} \dots i_N}^{N-k+1} = 1$  for some  $(i_{k+2}, \dots, i_N)$ . By this reasoning, variables of the form  $z_{00 i_{k+2} \dots i_N}^{N-k+1}$  can be assigned a value of one in the recovery problem only if  $w_{00}^2 = 1$ . However, variables  $z_{00 i_{k+2} \dots i_N}^{N-k+1}$  will be treated differently in the recovery procedure in that they can be assigned either zero or one independent of the value  $w_{00}^2$ . This increases the feasible set of the recovery problem, leading to a potentially better solution.

Let  $\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\}_{l_{k+1}=0}^{L_{k+1}}$  be an enumeration of indices  $\{l_k, i_{k+1}\}$  of  $w^2$  (or the first 2 indices of  $w^{N-k+1}$  constructed in equation (4.11)) such that

$$\begin{aligned} & w_{l_k(l_{k+1}), i_{k+1}(l_{k+1})}^2 = 1 \quad \text{for} \\ 5 \quad & (l_k(l_{k+1}), i_{k+1}(l_{k+1})) \neq (0, 0) \quad \text{and} \\ & (l_k(l_{k+1}), i_{k+1}(l_{k+1})) = (0, 0) \quad \text{for } l_{k+1}=0 \end{aligned}$$

regardless of whether  $w_{00}^2=1$  or not. To define the  $(N-k)$ -dimensional assignment problem that restores feasibility, let

$$\begin{aligned} c_{l_2 i_3 \dots i_N}^{N-1} &= c_{l_1(l_2) i_2(l_2) i_3 \dots i_N}^N \quad \text{for } k=1 \\ c_{l_{k+1} i_{k+2} \dots i_N}^{N-k} &= c_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} \\ &= c_{i_1(l_{2 \dots k+1}) i_2(l_{2 \dots k+1}) i_3(l_{3 \dots k+1}) \dots i_k(l_{kk+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^N \\ &\quad \text{for } k \geq 2 \text{ and } l_k = 0, \dots, L_k, \end{aligned} \quad (4.18)$$

where

$$l_{m \dots k+1} = l_m \circ l_{m+1} \circ \dots \circ l_k(l_{k+1}) \equiv l_m(l_{m+1}(\dots(l_k(l_{k+1}))) \dots) \quad (4.19)$$

10 for  $m = 2, \dots, k+1$  and where " $\circ$ " denotes function composition. For example,  $l_{23} = l_2(l_3)$  and  $l_{234} = l_2 \circ l_3(l_4) = l_2(l_3(l_4))$ .

Then the  $(N-k)$ -dimensional assignment problem that restores feasibility is

$$\begin{aligned}
& \text{Minimize} \quad \sum_{l_{k+1} i_{k+2} \dots i_N} C_{l_{k+1} i_{k+2} \dots i_N}^{N-k} Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} \\
& \text{Subject To} \quad \sum_{i_{k+2} \dots i_N} Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} = 1, \quad l_{k+1} = 1, \dots, L_{k+1}, \\
& \quad \sum_{i_{k+1} i_{k+3} \dots i_N} Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} = 1, \quad i_{k+2} = 1, \dots, M_{k+2}, \\
& \quad \sum_{l_k i_{k+1} \dots i_{p-1} i_{p+1} \dots i_N} Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} = 1 \\
& \quad \text{for } i_p = 1, \dots, M_p \text{ and } p = k+3, \dots, N-1, \\
& \quad \sum_{l_{k+1} i_{k+2} \dots i_{N-1}} Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} = 1, \quad i_N = 1, \dots, M_N, \\
& \quad Z_{l_{k+1} i_{k+2} \dots i_N}^{N-k} \in \{0, 1\} \text{ for all } l_{k+1}, i_{k+2}, \dots, i_N.
\end{aligned} \tag{4.20}$$

Let  $Y$  be an optimal or feasible solution to this  $(N-k)$ -dimensional assignment problem. The recovered feasible solution  $z^N$  is defined by

$$z_{i_1 i_2 i_3 \dots i_N}^N = \begin{cases} 1, & \text{if } i_1 = i_1(l_{2 \dots k+1}), i_2 = i_2(l_{2 \dots k+1}), \\ & i_3 = i_3(l_{2 \dots k+1}), \dots, i_k = i_k(l_{kk+1}), \\ & i_{k+1} = i_{k+1}(l_{k+1}) \text{ and } Y_{l_{k+1} i_{k+2} \dots i_N} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{4.21}$$

Said in a different way, the recovered feasible solution  $z^N$  corresponding to the multiplier set  $\{u^{k+2}, \dots, u^N\}$  is then defined by

$$z_{i_1(l_{2-k+1})i_2(l_{2-k+1})i_3(l_{3-k+1})\dots i_k(l_{kk+1})i_{k+1}(l_{k+1})i_{k+2}\dots i_N}^N = \begin{cases} 1, & \text{if } Y_{l_{k+1}i_{k+2}\dots i_N} = 1, \\ 0, & \text{otherwise,} \end{cases}$$

where  $l_{m\dots k+1}$  is defined in (4.19) and " $\circ$ " denotes function composition.

The next objective is to remove the assumption that all cost coefficients are defined and all zero-one variables are free to be assigned. We first note that the above construction of a reduced dimension assignment problem (4.20) is valid as long as all cost coefficients  $c^{N-k}$  are defined and all zero-one variables in  $z^{N-k}$  are free to be assigned.

Modifications are necessary for *sparse* problems. If the cost coefficient  $c_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\dots i_N}^{N-k+1}$  is well-defined and the zero-one variable  $z_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\dots i_N}^{N-k+1}$  is free to be assigned to zero or one, then define  $c_{l_{k+1}i_{k+2}\dots i_N}^{N-k} = c_{l_k(l_{k+1})i_{k+1}(l_{k+1})i_{k+2}\dots i_N}^{N-k+1}$  as in (4.18) with  $z_{l_{k+1}i_{k+2}\dots i_N}^{N-k}$  being free to be assigned. Otherwise,  $z_{l_{k+1}i_{k+2}\dots i_N}^{N-k}$  is preassigned to zero or the corresponding arc is not allowed in the feasible set of arcs. To ensure that a feasible solution exists, we now only need ensure that the variables  $z_{l_{k+1}0\dots 0}^{N-k}$  are free to be assigned for  $l_{k+1} = 0, 1, \dots, L_{k+1}$  with the corresponding cost coefficients being well-defined. (Recall that all

variables of the form  $z_{l_k 0 \dots 0}^{N-k+1}$  and  $z_{0 \dots 0 i_p 0 \dots 0}^{N-k+1}$  are free (to be assigned) and all coefficients of the form  $c_{l_k 0 \dots 0}^{N-k+1}$  and  $c_{0 \dots 0 i_p 0 \dots 0}^{N-k+1}$  are well-defined for  $l_k = 0, \dots, L_k$  and  $i_p = 0, \dots, M_p$  for  $p=k+1, \dots, N$ .) This is accomplished as follows: If the cost  
 5 coefficient  $c_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) 0 \dots 0}^{N-k+1}$  is well-defined and  $z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) 0 \dots 0}^{N-k+1}$  is free, then define  $c_{l_{k+1} 0 \dots 0}^{N-k} = c_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) 0 \dots 0}^{N-k+1}$  with  $z_{l_{k+1} 0 \dots 0}^{N-k}$  being free. Otherwise, since all variables of the form  $z_{l_k 0 \dots 0}^{N-k+1}$  and  $z_{0 i_{k+1} 0 \dots 0}^{N-k+1}$  are free to be assigned with the corresponding costs being well-defined, set  $c_{l_{k+1} 0 \dots 0}^{N-k} = c_{l_k(l_{k+1}) 0 0 \dots 0}^{N-k+1} + c_{0 i_{k+1}(l_{k+1}) 0 \dots 0}^{N-k+1}$ , where the  
 10 first term is omitted if  $l_k(l_{k+1})=0$  and the second, if  $i_{k+1}(l_{k+1})=0$ . For  $l_k(l_{k+1})=0$  and  $i_{k+1}(l_{k+1})=0$ , define  $c_{0 \dots 0}^{N-k} = c_{0 \dots 0}^{N-k+1}$ .

#### IV.3.4. The last step $k = N-2$ .

$$\begin{aligned} \Phi_2 = \text{Minimize} \quad & \sum_{l_{N-1}=0}^{L_{N-1}} \sum_{i_N=0}^{M_N} c_{l_{N-1} i_N}^2 z_{l_{N-1} i_N}^2 \equiv V_2(z^2) \\ \text{Subject To} \quad & \sum_{i_N=0}^{M_N} z_{l_{N-1} i_N}^2 = 1, \quad l_{N-1} = 1, \dots, L_{N-1}, \\ & \sum_{l_{N-1}=0}^{L_{N-1}} z_{l_{N-1} i_N}^2 = 1, \quad i_N = 1, \dots, M_N, \\ & z_{l_{N-1} i_N}^2 \in \{0, 1\} \text{ for all } l_{N-1}, i_N. \end{aligned} \tag{4.22}$$

The description of the algorithm ends with  $k = N-2$ . The  
 15 resulting recovery problem defined in (4.10) with  $k = N-2$  is

Let  $Y$  be an optimal or feasible solution to this two-dimensional assignment problem. The recovered feasible solution  $z^N$  is defined by

$$5 \quad z_{i_1 i_2 i_3 \dots i_N}^N = \begin{cases} 1, & \text{if } i_1 = i_1(l_{2 \dots N-1}), i_2 = i_2(l_{2 \dots N-1}), i_3 = i_3(l_{3 \dots N-1}), \dots \\ & i_{N-2} = i_{N-2}(l_{N-2N-1}), i_{N-1} = i_{N-1}(l_{N-1}) \text{ and } Y_{l_{N-1} i_N} = 1 \\ & \text{or if } (l_{N-1}, i_N) = (0, 0), \\ 0, & \text{otherwise.} \end{cases} \quad (4.23)$$

Said in a different way, the recovered feasible solution  $z^N$  is then defined by

$$z_{i_1(l_{2 \dots N-1}) i_2(l_{2 \dots N-1}) i_3(l_{2 \dots N-1}) \dots i_{N-2}(l_{N-2N-1}) i_{N-1}(l_{N-1}) i_N} = \begin{cases} 1, & \text{if } Y_{l_{N-1} i_N} = 1, \\ 0, & \text{otherwise.} \end{cases} \quad (4.24)$$

where  $l_{m \dots N-1}$  is defined in (4.19) and " $\circ$ " denotes function composition. To complete the description, let  
 10  $\{(l_{N-1}(l_N), i_N(l_N))\}_{l_N=0}^{l_N}$  be an enumeration of indices  $\{l_{N-1}, i_N\}$  of  $Y$  such that  $Y_{l_{N-1}(l_N), i_N(l_N)} = 1$  for  $(l_{N-1}(l_N), i_N(l_N)) \neq (0, 0)$  and  $(l_{N-1}(l_N), i_N(l_N)) = (0, 0)$  for  $l_N = 0$  regardless of whether  $Y_{00} = 1$  or not. Then the recovered feasible solution can be written as

$$15 \quad z_{i_1 i_2 i_3 \dots i_N}^N = \begin{cases} 1, & \text{if } i_1 = i_1(l_{2 \dots N}), i_2 = i_2(l_{2 \dots N}), i_3 = i_3(l_{3 \dots N}), \dots, \\ & i_{N-1} = i_{N-1}(l_{N-1N}), i_N = i_N(l_N); \\ 0, & \text{otherwise.} \end{cases} \quad (4.25)$$

#### IV.3.5. The Upper and Lower Bounds

The upper bound on the feasible solution is given by

$$V_N(\hat{z}^N) = \sum_{i_1 \dots i_N} c_{i_1 \dots i_N}^N \hat{z}_{i_1 \dots i_N}^N \quad (4.26)$$

and the lower by  $\phi_N(u^3, \dots, u^N)$  for any multiplier value  $(u^3, \dots, u^N)$ .

In particular, we have

5

$$\phi_N(u^3, \dots, u^N) \leq \phi_N(\bar{u}^3, \dots, \bar{u}^N) \leq V_N(\bar{z}^N) \leq V_N(\hat{z}^N), \quad (4.27)$$

where  $(u^3, \dots, u^N)$  is any multiplier value,  $(\bar{u}^3, \dots, \bar{u}^N)$  is a maximizer of  $\phi_N(u^3, \dots, u^N)$ ,  $\bar{z}^N$  is an optimal solution of the  
 10 multidimensional assignment problem (4.7) and  $\hat{z}^N$  is the recovered feasible solution defined by (4.25). Finally, we conclude with the observation that  $V_N(\hat{z}) = V_2(Y)$  where  $Y$  is the optimal solution of (4.23) used in the construction of  $\hat{z}$  in equations (4.24) - (4.26).

15

#### IV.3.6. Reuse of Multipliers

Since the most computationally expensive part of the algorithm occurs in the maximization of  $\phi_{N-k+1}$ , the development of algorithms that can make use of hot starts or multipliers  
 20 close to the optimal are fundamentally important for real-time speed. The purpose of this section is to demonstrate that the multiplier set obtained at stage  $k \geq 1$  provide good starting values for those obtained at step  $k+1$ .

**Theorem 4.4.** Let  $(u^3, \dots, u^N)$  denote a multiplier set obtained in the maximization of  $\Phi_N(u^3, \dots, u^N)$ . Then this multiplier set satisfies the string of inequalities

$$\Phi_N(u^3, \dots, u^N) \leq \Phi_{N-1}(u^4, \dots, u^N) \leq \dots \leq \Phi_3(u^N) \leq \Phi_2 \leq V_N(\bar{z}), \quad (4.28)$$

where after the first step in the maximization of  $\Phi_N$  the  
 5 multipliers are not changed in the remaining steps.  
 Furthermore, to improve this inequality, let  
 $(u^{2+k, N-k+1}, \dots, u^{N, N-k+1})$  denote a maximizer of  $\Phi_{N-k+1}(u^{k+2}, \dots, u^N)$ .  
 Then, we have

$$\begin{aligned} \Phi_{N-k+1}(u^{k+2}, \dots, u^N) &\leq \Phi_{N-k+1}(u^{2+k, N-k+1}, \dots, u^{N, N-k+1}) \\ &\leq \Phi_{N-k}(u^{3+k, N-k+1}, \dots, u^{N, N-k+1}) \\ &\leq \Phi_{N-k}(u^{3+k, N-k}, \dots, u^{N, N-k}) \\ &\leq \dots \leq \Phi_3(u^N) \leq \Phi_2 \leq V_N(\hat{z}). \end{aligned} \quad (4.29)$$

**Proof:** Suppose we have a value of the multipliers  $(u^{k+2}, \dots, u^N)$   
 10 obtained in the maximization of

$$\Phi_{N-k+1}(u^{k+2}, \dots, u^N) \equiv \text{Minimize } \phi_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) \quad (4.30)$$

$$\begin{aligned} \text{Subject To } \sum_{i_{k+1} i_{k+2} \dots i_N} z_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} &= 1, \quad l_k = 1, \dots, L_k, \\ \sum_{l_k i_{k+2} \dots i_N} z_{l_k i_{k+1} i_{k+2} \dots i_N}^{N-k+1} &= 1, \quad i_{k+1} = 1, \dots, M_{k+1}, \end{aligned}$$

where



$$\begin{aligned}
& \phi_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) \\
&= \sum_{l_k i_{k+1} \dots i_N} c_{l_k i_{k+1} \dots i_N}^{N-k+1} z_{l_k i_{k+1} \dots i_N}^{N-k+1} \\
&+ \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} \left[ \sum_{l_k i_{k+1} \dots i_{p-1} i_{p+1} \dots i_N} z_{l_k i_{k+1} \dots i_N}^{N-k+1} - 1 \right] \quad (4.31) \\
&= \sum_{l_k i_{k+1} \dots i_N} \left[ c_{l_k i_{k+1} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \right] z_{l_k i_{k+1} \dots i_N}^{N-k+1} - \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p.
\end{aligned}$$

These need not be the maximizer; however, we do assume that we have solved the above minimization problem optimally to evaluate  $\phi_{N-k+1}(u^{k+2}, \dots, u^N)$ . Just as in the definition of the recovery problem discussed earlier, let

5  $\{(l_k(l_{k+1}), i_{k+1}(l_{k+1}))\}_{l_{k+1}=0}^{L_{k+1}}$  be an enumeration of indices  $\{l_k, i_{k+1}\}$  of the optimal solution  $w^2$  of problem (4.10) (or the first 2 indices of the solution  $w^{N-k+1}$  of the relaxed problem (4.8) constructed in equation (4.11)) such that  $w_{l_k(l_{k+1}), i_{k+1}(l_{k+1})}^2 = 1$  for  $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) \neq (0, 0)$  and  $(l_k(l_{k+1}), i_{k+1}(l_{k+1})) = (0, 0)$  for  $l_{k+1}=0$

10 regardless of whether  $w_{00}^2=1$  or not.

Then, the final value of the objective function can be expressed as

$$\Phi_{N-k+1}(u^{k+2}, \dots, u^N) \equiv$$

$$\text{Minimize } \phi_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) \quad (4.32)$$

$$\text{Subject to } \sum_{i_{k+2} \dots i_N} z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k-1} = 1, \quad l_{k+1} = 1, \dots, L_{k+1},$$

where

$$\phi_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) = \quad (4.33)$$

$$\sum_{l_{k+1} i_{k+2} \dots i_N} \left[ C_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \right] z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} \\ - \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p.$$

If now another constrain set, say the  $(k+2)^{\text{th}}$  set, is added to  
5 this problem, one has

$$\tilde{\Phi}_{N=k+1}(u^{k+2}, \dots, u^N) \equiv \text{Minimize } \tilde{\phi}_{N-k+1}(z^{N-k+1}; u^{k+2}, \dots, u^N) \equiv \quad (4.34)$$

$$\text{Minimize } \sum_{l_{k+1} i_{k+2} \dots i_N} \left\{ C_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} + \sum_{p=k+2}^N u_{i_p}^p \right\} z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} \\ - \sum_{p=k+2}^N \sum_{i_p=0}^{M_p} u_{i_p}^p$$

$$\text{Subject to } \sum_{i_{k+2} \dots i_N} z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} = 1, \quad l_{k+1} = 1, \dots, L_{k+1}, \\ \sum_{l_{k+1} i_{k+3} \dots i_N} z_{l_k(l_{k+1}) i_{k+1}(l_{k+1}) i_{k+2} \dots i_N}^{N-k+1} = 1, \quad i_{k+2} = 1, \dots, M_{k+2}.$$

Since the constraint  $\sum_{i_{k+1} i_{k+2} \dots i_N} z_{i_{k+1} (i_{k+1}) i_{k+1} (i_{k+1}) i_{k+2} \dots i_N}^{N-k+1} = 1$  for  $i_{k+2}=1, \dots, M_{k+2}$  is now imposed, the feasible region is smaller, so that one has

$$\begin{aligned} \Phi_{N-k+1}(u^{k+2}, \dots, u^N) &\leq \tilde{\Phi}_{N-k+1}(u^{k+2}, \dots, u^N) \\ &\equiv \tilde{\Phi}_{N-k+1}(u^{k+3}, \dots, u^N) \\ &\equiv \Phi_{N-k+1}(u^{k+3}, \dots, u^N), \end{aligned}$$

where the fact that  $\tilde{\Phi}_{N-k+1}$  does not depend on the multiplier set  $u^{k+2}$  due the added constraint set. Also, the last equivalence follows from the fact that  $\Phi_{N-k+1}(u^{k+3}, \dots, u^N)$  is the relaxed problem (with  $k$  replaced by  $k+1$ ) for the recovery problem in step  $k$  of the above algorithm. Continuing this way, one can compute  $(u^3, \dots, u^N)$  at the first step ( $k=1$ ), fix them thereafter, and perform no optimization at the subsequent steps to obtain

$$\Phi_N(u^3, \dots, u^N) \leq \Phi_{N-1}(u^4, \dots, u^N) \leq \dots \leq \Phi_3(u^N) \leq \Phi_2 \equiv V_N(\hat{Z}),$$

where  $\Phi_2$  is defined in (4.22).

To explain how to improve this inequality, let  $(u^{2+k, N-k+1}, \dots, u^{N, N-k+1})$  denote a maximizer of  $\Phi_{N-k+1}(u^{k+2}, \dots, u^N)$ . Then by the same reasoning one has the result as stated in the theorem.

V. SUMMARY OF THE LAGRANGIAN RELAXATION PROBLEM

Given the multidimensional assignment problem

$$\begin{aligned}
 &\text{Minimize} && \sum_{i_1 \dots i_N} c_{i_1 \dots i_N} z_{i_1 \dots i_N} \\
 &\text{Subject To} && \sum_{i_2 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_1 = 1, \dots, M_1), \\
 &&& \sum_{i_1 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_2 = 1, \dots, M_2), \\
 &&& \sum_{i_1 \dots i_{p-1} i_{p+1} \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_p = 1, \dots, M_p \text{ and } p = 2, \dots, N-1), \\
 &&& \sum_{i_1 i_2 \dots i_{N-1}} z_{i_1 \dots i_N} = 1, \quad (i_N = 1, \dots, M_N), \\
 &&& z_{i_1, \dots, i_N} \in \{0, 1\} \text{ for all } i_1, \dots, i_N,
 \end{aligned} \tag{5.1}$$

where  $c_{0 \dots 0}^N$  is arbitrarily defined to be zero and is included for notational convenience and where the superscript  $N$  on both  $c$  and  $z$  is not an exponent, but denotes the dimension of the  
 5 subscripts and the assignment problem as stated in (3.5). In relaxed and recovery problems  $c_{0 \dots 0}^N$  need not be zero! In this problem, we assume that all zero-one variables  $z_{i_1 \dots i_N}^N$  with precisely one nonzero index are free to be assigned and that  
 10 the corresponding cost coefficients are well-defined. (This is a valid assumption in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking, partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical*

Computer Science, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. 5 *Computational Optimization and Applications*, 3:27-57, 1994).) Although not required, these cost coefficients with exactly one nonzero index can be translated to zero by cost shifting (A.B. Poore and N. Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising from 10 multitarget tracking. *SIAM Journal of Optimization*, 3, No. 3:544-563, 1993) without changing the optimal assignment.

Having explained many of the relaxation features, it is now appropriate to present the Lagrangian relaxation algorithm, which iteratively constructs a feasible solution to 15 the  $N$ -dimensional assignment problem (5.1).

**Algorithm 5.1** (*Lagrangian Relaxation Algorithm*) To construct a high quality feasible solution, denoted by  $\hat{z}^N$ , of the assignment problem (5.1), proceed as follows:

0. Initialize the multipliers  $(u^{k+2}, \dots, u^N)$ , e.g., 20  $(u^{k+2}, \dots, u^N) = (0, \dots, 0)$ .

**For**  $k=1, \dots, N-2$ , **do**

1. For the Lagrangian relaxed problem (4.8) from the problem (4.7) by relaxing the last  $(N-k-1)$  sets of constraints.

2. Use a nonsmooth optimization technique to solve

$$\begin{aligned} \text{Maximize } \{ \phi_{N-k+1} | u^p \in \mathbb{R}^{M_p+1} \text{ for } p=k+2, \dots, N \\ \text{with } u_0^p=0 \text{ being fixed} \}, \end{aligned} \quad (5.2)$$

where  $\phi_{N-k+1}(u^{k+2}, \dots, u^N)$  is defined by equation (4.8). The algorithm in Section IV.3.2 provides one method for computing a function value and a subgradient out of the subdifferential at  $(u^{k+2}, \dots, u^N)$ , as required in most nonsmooth optimization techniques.

3. Given an approximate or optimal maximizer of (5.2), say  $(\hat{u}^{k+2}, \dots, \hat{u}^N)$ , let  $\hat{w}^2$  denote the optimal solution of the two-dimensional assignment problem (4.10) corresponding to this maximizer of  $\phi_{N-k+1}(u^{k+2}, \dots, u^N)$ .
4. Formulate the recovery  $(N-k)$ -dimensional problem (4.19), modified as discussed at the end of subsection IV.3.3 for sparse problems. At this stage,  $\hat{z}^N$ , as defined in (4.21), contains the alignment of the indices  $\{i_1, \dots, i_{k+1}\}$ .
- 15 Formulate the final two-dimensional problem (4.22) and complete the final recovered solution  $\hat{z}^N$  as in (4.25).

To explain the lower and upper bounds, let  $\phi_N$  be as defined in (4.8) with  $k=1$ , let  $V_N(z^N)$  be the objective function value of the  $N$ -dimensional assignment problem in equation (3.5) corresponding to a feasible solution  $z^N$  of the

constraints in (3.5), and let  $\bar{z}^{N-k+1}$  be an optimal solution of (3.5). Then,

$$\Phi_N(u^3, \dots, u^N) \leq V_N(\bar{z}^N) \leq V_N(\hat{z}^N) \quad (5.3)$$

is the desired inequality.

5

#### COMMENTS:

1. Step 2 is the computational part of the algorithm. Evaluating  $\Phi_{N-k+1}$  and computing a subgradient used in the search procedure requires 99% of the computing time in the algorithm. This part uses two-dimensional assignment algorithms, a search over a large number of indices, and a nonsmooth optimization algorithm. It is the second part (the search) that consumes 99% of the computational time and this is almost entirely parallelizable.
2. In track maintenance, the warm starts for the initial multipliers for the first step are available. For the relaxation procedure, initial multipliers are available in step 2 from the prior step in the algorithm.
3. There are many variations on the above algorithm. For example, one can compute a good solution on the first pass ( $k=1$ ) and not perform the optimization in (2) thereafter. This yields a great solution. Thus one can

continue the optimization at the first pass, and immediately compute quality feasible solutions to the problem.

5

# VI. LAGRANGIAN RELAXATION ALGORITHM FOR THE THREE-DIMENSIONAL ALGORITHM

In this section, we illustrate the relaxation algorithm for the three-dimensional assignment problem. Having discussed the general relaxation scheme,

$$\begin{aligned}
 &\text{Minimize} && \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} C_{i_1 i_2 i_3} z_{i_1 i_2 i_3} \\
 &\text{Subject To} && \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3} = 1, \quad i_1 = 1, \dots, M_1, \\
 &&& \sum_{i_1=0}^{M_1} \sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3} = 1, \quad i_2 = 1, \dots, M_2, \\
 &&& \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} z_{i_1 i_2 i_3} = 1, \quad i_3 = 1, \dots, M_3, \\
 &&& z_{i_1 i_2 i_3} \in (0, 1) \text{ for all } i_1 i_2 i_3.
 \end{aligned} \tag{6.1}$$

10

To ensure that a feasible solution of (6.1) always exists for a sparse problem, all variables with exactly one nonzero index (i.e., variables of the form  $z_{i_1 0 0}$ ,  $z_{0 i_2 0}$ , and  $z_{0 0 i_3}$  for  $i_p = 1, \dots, M_p$  and  $p = 1, 2, 3$  are assumed free to be assigned with the corresponding cost coefficients being well-defined). This assumption is valid in the tracking environment (A.B. Poore. Multidimensional assignments and multitarget tracking,



partitioning data sets, I.J. Cox, P. Hansen, and B. Julesz, editors, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, Providence, R.I., 19:169-198, 1995; A.B. Poore. Multidimensional  
 5 assignment formulation of data association problems arising from multitarget tracking and multisensor data fusion. *Computational Optimization and Applications*, 3:27-57, 1994).

#### VI.1. The Lagrangian Relaxed Assignment Problem

10 The three-dimensional assignment problem (6.1) has three sets of constraints and one can describe the relaxation by relaxing any of the three sets of constraints, the description here is based on relaxing the last set of constraints. A  $(M_3+1)$ -dimensional multiplier vector (i.e.,  $u^3 \in \mathbb{R}^{M_3+1}$ )  
 15 associated with the 3<sup>rd</sup> constraint set will be denoted by  $u^3 = (u_0^3, u_1^3, \dots, u_{M_3}^3)^T$  with  $u_0^3 \equiv 0$  being fixed for notational convenience. (The zero multiplier  $u_0^3 \equiv 0$  is used for notational convenience.) Now, the three-dimensional assignment problem (6.1) is relaxed to a two-dimensional assignment problem by  
 20 incorporating last set of constraints into the objective function via the Lagrangian. Although any constraint set can be relaxed, we choose the last set of constraints for convenience. The relaxed problem is

(6.2)

$$\begin{aligned}
\Phi_3(u^3) &\equiv \text{Minimize } \phi_3(z^3; u^3) \equiv \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} C_{i_1 i_2 i_3}^3 z_{i_1 i_2 i_3}^3 \\
&\quad + \sum_{i_3=0}^{M_3} u_{i_3}^3 \left[ \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} z_{i_1 i_2 i_3}^3 - 1 \right] \\
&\equiv \text{Minimize } \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} \left[ C_{i_1 i_2 i_3}^3 + u_{i_3}^3 \right] z_{i_1 i_2 i_3}^3 - \sum_{i_3=0}^{M_3} u_{i_3}^3 \\
\text{Subject to } &\sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3}^3 = 1, \quad i_1 = 1, \dots, M_1, \\
&\sum_{i_1=0}^{M_1} \sum_{i_3=0}^{M_3} z_{i_1 i_2 i_3}^3 = 1, \quad i_2 = 1, \dots, M_2.
\end{aligned}$$

One of the major steps in the algorithm is the maximization of  $\Phi_3(u^3)$  with respect to the multiplier vector  $u^3$ . It turns out that  $\Phi_3$  is a concave, continuous, and piecewise affine function of the multiplier vector  $u^3$ , so that the maximization of  $\Phi_3$  is a problem of nonsmooth optimization. Since many of these algorithms require a function value and a subgradient of  $\Phi_3$  at any required multiplier value ( $u^3$ ), we address this problem in the next subsection. We note, however, that there are other ways to maximize  $\Phi_3$  and the next subsection addresses but one such method.

## VI.2. Properties Lagrangian Relaxed Assignment Problem

For a function evaluation of  $\Phi_3$ , we show that an optimal (or suboptimal) solution of this relaxed problem (6.2) can be constructed from that of a two-dimensional assignment problem.

Then, the nonsmooth characteristics of  $\Phi_3$  are addressed, followed by a method for computing a subgradient.

**Evaluation of  $\Phi_3$ .** Define for each  $(i_1, i_2)$  an index  $j_3 = j_3(i_1, i_2)$  and a new cost function  $c_{i_1 i_2}^2$  by:

$$\begin{aligned} 5 \quad j_3 &\equiv j_3(i_1, i_2) = \arg \min \left\{ c_{i_1 i_2 i_3}^3 + u_{i_3}^3 \mid i_3 = 0, 1, \dots, M_3 \right\}, \\ c_{i_1 i_2}^2 &= c_{i_1 i_2 j_3(i_1 i_2)}^3 + u_{j_3}^3 \text{ for } (i_1, i_2) \neq (0, 0), \\ c_{00}^2 &= \sum_{i_3=0}^{M_3} \text{Minimum} \left\{ 0, c_{00 i_3}^3 + u_{i_3}^3 \right\}. \end{aligned} \quad (6.3)$$

Given an index pair  $(i_1, i_2)$ ,  $j_3$  need not be unique, resulting in the potential generation of several subgradients (6.11). (We further discuss this issue at the end of the Subsection.)

10 Now, define

$$\begin{aligned} \hat{\Phi}_3(u^3) = \text{Minimize} \quad \hat{\Phi}_3(z^2; u^3) &\equiv \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} c_{i_1 i_2}^2 z_{i_1 i_2}^2 - \sum_{i_3=0}^{M_3} u_{i_3}^3. \\ \text{Subject To} \quad \sum_{i_2=0}^{M_2} z_{i_1 i_2}^2 &= 1, \quad i_1 = 1, \dots, M_1, \\ \sum_{i_1=0}^{M_1} z_{i_1 i_2}^2 &= 1, \quad i_2 = 1, \dots, M_2, \\ z_{i_1 i_2}^2 &\in \{0, 1\} \text{ for all } i_1, i_2. \end{aligned} \quad (6.4)$$

As an aside, two observations are in order. The first is that the search procedure needed for the computation of the relaxed cost coefficients in (6.3) is the most computationally intensive part of the entire relaxation algorithm. The second

is that a feasible solution  $z^3$  of a sparse problem (6.1) yields a feasible solution  $z^2$  of (6.4) via the construction

$$z_{i_1 i_2}^2 = \begin{cases} 1, & \text{if } z_{i_1 i_2, i_3}^3 = 1 \text{ for some } (i_1, i_2, i_3), \\ 0, & \text{otherwise.} \end{cases}$$

- 5 Thus the two-dimensional assignment problem (6.4) has feasible solutions other than those with exactly one nonzero index if the original problem (6.1) does.

The following Theorem 6.1 states that an optimal solution of (6.2) can be computed from that of (6.4). Furthermore, if  
10 the solution of either of these two problems is  $\epsilon$ -optimal, then so is the other. The converse is contained in Theorem 6.2.

**Theorem 6.1.** Let  $w^2$  be a feasible solution to the two-dimensional assignment problem (6.4) and define  $w^3$  by

$$\begin{aligned} 15 \quad w_{i_1 i_2 i_3}^3 &= w_{i_1 i_2}^2 && \text{if } i_3 = j_3 \text{ and } (i_1, i_2) \neq (0, 0), \\ w_{i_1 i_2 i_3}^3 &= 0 && \text{if } i_3 \neq j_3 \text{ and } (i_1, i_2) \neq (0, 0), \\ w_{00 i_3}^3 &= 1 && \text{if } c_{00 i_3}^3 + u_{i_3}^3 \leq 0, \\ w_{00 i_3}^3 &= 0 && \text{if } c_{00 i_3}^3 + u_{i_3}^3 > 0. \end{aligned} \tag{6.5}$$

Then  $w^3$  is a feasible solution of the Lagrangian relaxed problem (6.2) and  $\phi_3(w^3; u^3) = \hat{\phi}_3(w^2; u^3)$ . If, in addition,  $w^2$  is

optimal for the two-dimensional problem, then  $w^3$  is an optimal solution of the relaxed problem and  $\phi_3(u^3) = \hat{\phi}_3(u^3)$ .

With the exception of one equality being converted to an inequality, the following theorem is a converse of Theorem 6.1.

**Theorem 6.2.** Let  $w^3$  be a feasible solution to problem (6.2) and define  $w^2$  by

$$w_{i_1 i_2}^2 = \sum_{i_3=0}^{M_3} w_{i_1 i_2 i_3}^3 \text{ for } (i_1, i_2) \neq (0, 0), \quad (6.6)$$

$$w_{00}^2 = 0 \text{ if } (i_1, i_2) = (0, 0) \text{ and } c_{00 i_3}^3 + u_{i_3}^3 > 0 \text{ for all } i_3,$$

$$w_{00}^2 = 1 \text{ if } (i_1, i_2) = (0, 0) \text{ and } c_{00 i_3}^3 + u_{i_3}^3 \leq 0 \text{ for some } i_3.$$

Then  $w^2$  is a feasible solution of the problem (6.4) and  $\phi_3(w^3; u^3) \geq \hat{\phi}_3(w^2; u^3)$ . If, in addition,  $w^3$  is optimal for (6.2), then  $w^2$  is an optimal solution of (6.4),  $\phi_3(w^3; u^3) = \hat{\phi}_3(w^2; u^3)$  and  $\phi_3(u^3) = \hat{\phi}_3(u^3)$ .

#### An Algorithm for Computing $\phi_3$ and a Subgradient.

Given  $u^3$  the problem of computing  $\phi_3(u^3)$  and a subgradient of  $\phi_3(u^3)$  can be summarized as follows.

1. Starting with problem (6.1), form the relaxed problem (6.2).
2. To solve (6.2) optimally, define the two-dimensional assignment problem (6.4) via the transformation (6.3).
- 5 3. Solve the two-dimensional problem (6.4) optimally.
4. Reconstruct the optimal solution, say  $\hat{w}^3$ , of (6.2) via equation (6.5) as in Theorem 6.1.
5. Then

$$\begin{aligned} \Phi_3(u^3) \equiv \phi_3(\hat{w}^3; u^3) = & \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} c_{i_1 i_2 i_3}^3 \hat{w}_{i_1 i_2 i_3}^3 + \\ & \sum_{i_3=0}^{M_3} u_{i_3}^3 \left[ \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \hat{w}_{i_1 i_2 i_3}^3 - 1 \right]. \end{aligned} \quad (6.7)$$

- 10 6. A subgradient is given by substituting  $\hat{w}^3$  into the objective function (6.2), erasing the minimum sign, and taking the partial with respect to  $u_{i_3}^3$ . The result is

$$g_{i_3}^3 = \frac{\partial \Phi_3(u^3)}{\partial u_{i_3}^3} = \left[ \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \hat{w}_{i_1 i_2 i_3}^3 - 1 \right] \text{ for } i_3 = 1, \dots, M_3. \quad (6.8)$$

### VI.3. The Recovery Procedure

- 15 The next objective is to explain a *recovery procedure*, i.e., given a feasible (optimal or suboptimal) solution  $w^2$  of (6.4) (or  $w^3$  of (6.2) constructed in Theorem 6.1), generate a feasible solution  $z^3$  of equation (6.1) which is

close to  $w^2$  in a sense to be specified. We first assume that no variables in (6.1) are preassigned to zero; this assumption will be removed shortly. The difficulty with the solution  $w^3$  in Theorem 6.1 is that it need not satisfy the

5 third set of constraints in (6.1). (Note, however, that if  $w^2$  is an optimal solution for (6.4) and  $w^3$ , as constructed in Theorem 6.1, satisfies the relaxed constraints, then  $w^3$  is optimal for (6.1).) The recovery procedure described here is designed to preserve the 0-1 character of the solution  $w^2$  of

10 (6.4) as far as possible: If  $w_{i_1 i_2}^2 = 1$  and  $i_1 \neq 0$  or  $i_2 \neq 0$ , the corresponding feasible solution  $z^3$  of (6.1) is constructed so that  $z_{i_1 i_2 i_3}^3 = 1$  for some  $(i_1, i_2, i_3)$ . By this reasoning, variables of the form  $z_{00 i_3}^3$  can be assigned a value of one in the recovery problem only if  $w_{00}^2 = 1$ . However, variables  $z_{00 i_3}^3$  will

15 be treated differently in the recovery procedure in that they can be assigned either zero or one independent of the value  $w_{00}^2$ . This increases the feasible set of the recovery problem, leading to a potentially better solution.

Let  $\{(i_1(l_2), i_2(l_2))\}_{l_2=0}^{L_1}$  be an enumeration of indices  $\{i_1, i_2\}$

20 of  $w^2$  (or the first 2 indices of  $w^3$  constructed in equation (6.5)) such that  $w_{i_1(l_2), i_2(l_2)}^2 = 1$  for  $(i_1(l_2), i_2(l_2)) \neq (0, 0)$  and  $(i_1(l_2), i_2(l_2)) = (0, 0)$  for  $l_2 = 0$  regardless of whether  $w_{00}^2 = 1$  or

not. (The latter can only improve the quality of the feasible solution.)

Next to define the two-dimensional assignment problem that restores feasibility, let

$$5 \quad c_{l_2 i_3}^2 = c_{i_1(l_2) i_2(l_2) i_3}^3 \text{ for } l_2 = 0, \dots, L_1. \quad (6.9)$$

Then the two-dimensional assignment problem that restores feasibility is

$$\text{Minimize } \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_2} c_{l_2 i_3}^2 z_{l_2 i_3}^2 \quad (6.10)$$

$$\begin{aligned} \text{Subject to } & \sum_{i_3=0}^{M_2} z_{l_2 i_3}^2 = 1, \quad l_2 = 1, \dots, L_2, \\ & \sum_{l_2=0}^{L_2} z_{l_2 i_3}^2 = 1, \quad i_3 = 1, \dots, M_3, \\ & z_{l_2 i_3}^2 \in \{0, 1\} \text{ for all } l_2, i_3. \end{aligned}$$

10        The next objective is to remove the assumption that all cost coefficients are defined and all zero-one variables are free to be assigned. We first note that the above construction of a reduced two-dimensional assignment problem (6.11) is valid as long as all cost coefficients  $c^2$  are defined  
15 and all zero-one variables in  $z^2$  are free to be assigned. Modifications are necessary for *sparse* problems. If the cost coefficient  $c_{i_1(l_2) i_2(l_2) i_3}^3$  is well-defined and the zero-one variable  $z_{i_1(l_2) i_2(l_2) i_3}^3$  is free to be assigned to zero or one, then



define  $c_{l_2 i_3}^2 = c_{i_1(l_2) i_2(l_2) i_3}^3$  as in (6.9) with  $z_{i_1(l_2) i_2(l_2) i_3}^3$  being free to be assigned. Otherwise,  $z_{l_2 i_3}^2$  is preassigned to zero or the corresponding arc is not allowed in the feasible set of arcs. To ensure that a feasible solution exists, we now only need

5 ensure that the variables  $z_{l_2 0}^2$  are free to be assigned for  $l_2 = 0, 1, \dots, L_1$  with the corresponding cost coefficients being well-defined. Recall that all variables of the form  $z_{i_1 00}^3, z_{0 i_2 0}^3$  and  $z_{00 i_3}^3$  are free (to be assigned) and all coefficients of the corresponding form  $c_{i_1 00}^3, c_{0 i_2 0}^3$  and  $c_{00 i_3}^3$  need to be defined. This

10 is accomplished as follows: if the cost coefficient  $c_{i_1(l_2) i_2(l_2) 0}^3$  is well-defined and  $z_{i_1(l_2) i_2(l_2) 0}^3$  is free, then define  $c_{l_2 0}^2 = c_{i_2(l_2) i_2(l_2) 0}^3$  with  $z_{l_2 0}^2$  being free. Otherwise, since all variables of the form  $z_{i_1 0}^3$  and  $z_{0 i_2 0}^3$  are free to be assigned with the corresponding costs being well-defined, set

15  $c_{l_2 0}^2 = c_{i_1(l_2) 00}^3 + c_{0 i_2(l_2) 0}^3$ , where the first term is omitted if  $i_1(l_2) = 0$  and the second, if  $i_2(l_2) = 0$ . For  $i_1(l_2) = 0$  and  $i_2(l_2) = 0$ , define  $c_{00}^2 = c_{000}^3$ .

#### VI.4. The final recovery problem

The recovery problem for the case  $N = 3$  is

$$\begin{aligned}
 & \text{Minimize} \quad \sum_{l_2=0}^{L_2} \sum_{i_3=0}^{M_3} c_{l_2 i_3}^3 z_{l_2 i_3}^2 \\
 & \text{Subject to} \quad \sum_{i_3=0}^{M_3} z_{l_2 i_3}^2 = 1, \quad l_2 = 1, \dots, L_2, \\
 & \quad \sum_{l_2=0}^{L_2} z_{l_2 i_3}^2 = 1, \quad l_3 = 1, \dots, M_3, \\
 & \quad z_{l_2 i_3}^2 \in \{0, 1\} \text{ for all } l_2, i_3.
 \end{aligned} \tag{6.11}$$

Let  $Y$  be an optimal or feasible solution to this two-dimensional assignment problem. The recovered feasible solution  $z^3$  is defined by

$$z_{i_1 i_2 i_3}^3 = \begin{cases} 1, & \text{if } i_1 = i_1(l_2), i_2 = i_2(l_2) \text{ and } Y_{l_2 i_3} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{6.12}$$

Said in another way, let  $\{(l_2(l_3), i_3(l_3))\}_{l_3=0}^{L_3}$  be an enumeration of indices  $\{l_2, i_3\}$  of  $Y$  such that  $Y_{l_2(l_3), i_3(l_3)} = 1$  for  $(l_2(l_3), i_3(l_3)) \neq (0, 0)$  and  $(l_2(l_3), i_3(l_3)) = (0, 0)$  for  $l_3 = 0$  regardless of whether  $Y_{00} = 1$  or not. Then the recovered feasible solution can be written as

$$z_{i_1 i_2 i_3}^3 = \begin{cases} 1, & \text{if } i_1 = i_1(l_{12}), i_2 = i_2(l_{12}), i_3 = i_3(l_3), \\ 0, & \text{otherwise.} \end{cases} \tag{6.13}$$

The upper bound on the feasible solution is given by

$$V_3(z^3) = \sum_{i_1=0}^{M_1} \sum_{i_2=0}^{M_2} \sum_{i_3=0}^{M_3} c_{i_1 i_2 i_3}^3 z_{i_1 i_2 i_3}^3 \tag{6.14}$$

and the lower by  $\phi_3(u^3)$  for any multiplier value  $(u^3)$ .

In particular, we have

$$\Phi_3(u^3) \leq \Phi_3(\bar{u}^3) \leq V_3(\bar{z}^3) \leq V_3(\hat{z}^3),$$

where  $u^3$  is any multiplier value,  $\bar{u}^3$  is a maximizer of  $\Phi_3(u^3)$ ,  $\bar{z}^3$  is an optimal solution of the multidimensional assignment  
5 problem and  $\hat{z}^3$  is the recovered feasible solution defined by  
(6.13).

#### VII. OTHER RELAXATIONS FOR THE MULTIDIMENSIONAL ASSIGNMENT PROBLEM

In this section, we briefly describe other possible  
10 relaxations and their implications. These include linear  
programming relaxations and the corresponding duals.

Recall from Section II that one starts with the  
definition of the zero-one variable  $z_{i_1 \dots i_N} = 1$  and cost  
coefficients to form the problem

$$\begin{aligned} & \text{Minimize } \sum_{i_1 \dots i_N} c_{i_1 \dots i_N} z_{i_1 \dots i_N} \\ & \text{Subject To } \sum_{i_2 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_1 = 1, \dots, M_1), \\ & \quad \sum_{i_1 i_3 \dots i_N} z_{i_1 \dots i_N} = 1 \quad (i_2 = 1, \dots, M_2), \\ & \quad \sum_{i_1 \dots i_{p-1} i_{p+1} \dots i_N} z_{i_1 \dots i_N} = 1 \\ & \quad \quad (i_p = 1, \dots, M_p \text{ and } p = 2, \dots, N-1), \\ & \quad \sum_{i_1 i_2 \dots i_{N-1}} z_{i_1 \dots i_N} = 1 \quad (i_N = 1, \dots, M_N), \\ & \quad z_{i_1 \dots i_N} \in \{0, 1\} \text{ for all } i_1, \dots, i_N, \end{aligned} \tag{7.1}$$

where  $c_{0...0}$  is arbitrarily defined to be zero. Here, each group of sums in the constraints represents the fact that each non-dummy report occurs exactly once in a "track of data". One can modify this formulation to include *multiassignments* of one, some, or all the actual reports. The assignment problem (3.5) is changed accordingly. For example, if  $z_{i_k}^k$  is to be assigned no more than, exactly, or no less than  $n_{i_k}^k$  times, then the "=1" in the constraint (3.5) is changed to " $\leq, =, \geq n_{i_k}^k$ ," respectively. Modifications for group tracking and multiresolution features of the surveillance region will be addressed in future work. In making these changes, one must pay careful attention to the independence assumptions that need not be valid in many applications.

Again, the recent work of Poore and Drummond (A.B. Poore and O.E. Drummond. Track initiation and maintenance using multidimensional assignment problems. In D.W. Hearn, W.W. Hager, and P.M. Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.) significantly extends the formulation of the track maintenance and initiation to new approaches. The discussions of this section apply equally to those formulations.

#### VII.1. The Linear Programming Relaxation

In the linear programming relaxation, one replaces the zero-one variables constraints

$$z_{i_1 \dots i_N} \in \{0, 1\} \text{ for all } i_1, \dots, i_N \quad (7.2)$$

with the constraint

$$0 \leq z_{i_1 \dots i_N} \leq 1 \text{ for all } i_1, \dots, i_N. \quad (7.3)$$

Then, the problem (7.1) can be formulated as a linear programming problem with the constraint (7.2) in (7.1) replaced by (7.3) with a special block structure to which the Dantzing-Wolfe decomposition is applicable. Of course, after solving this problem, one must now recover the zero-one character of the variables in (7.1) and there are many ways to do this, such as using the two-dimensional assignment problems. Commercial software is also available.

## VII.2. The Linear Programming Dual and Partial Lagrangian Relaxations

Given the linear programming relaxation, one can formulate the dual problem or the partial Lagrangian relaxation duals with respect to any number of constraints. In particular, this is precisely what is done in Section 3 on the Lagrangian relaxation algorithm presented. The much broader class of algorithms provided in the US Patent 5537119 (Aubrey B. Poore, Jr., Method and System for Tracking Multiple Regional Objects by Multi-Dimensional Relaxation, US Patent Number 5537119,

filed 14 March 1995, issue date of 16 July 1996) can be modified to remove the zero-one character when one relaxes  $M$  sets of constraints to an  $(N-M)$ -dimensional problem and recovers with an  $(N-M+1)$ -dimensional problem. This avoids  
 5 the problems associated with the NP-hard  $(N-M)$ - and  $(N-M+1)$ -dimensional problems. However, to restore the zero-one character, one can do it sequentially with an assignment problem or with one of the many zero-one rounding techniques. These formulations are easy to work out and thus the details  
 10 are omitted.

The complete dual problem is another way of solving the LP problem and may indeed be more efficient in certain applications. In addition, the solution of this dual problem may provide excellent initial approximations to the  
 15 multipliers for Lagrangian relaxations.

#### VIII. HOT STARTS FOR TRACK MAINTENANCE AND INITIATION: BUNDLE MODIFICATIONS

Thus reuse of multipliers and the first proof that this  
 20 reuse is actually valid was presented in this section. The reuse in track maintenance is demonstrated and discussed in the work of Poore and Drummond (A.B. Poore and O.E. Drummond. Track initiation and maintenance using multidimensional assignment problems. In D.W. Hearn, W.W. Hager, and P.M.

Pardalos, editors, *Network Optimization*, volume 450, pages 407-422, Boston, 1996. Kluwer Academic Publishers B.V.). The only item left is the modification of the bundle of subgradients for the use with the multiplier values as one goes through the recovery problem as in Section IV.3.6 or as one moves the window in track maintenance. It is this aspect of the nonsmooth optimization that adds an order of magnitude to the speed of the algorithms. This work is based on both a mathematical proof as in Section IV.3.6 as well as computational experience and heuristics.

#### IX. ADAPTIVE AND VARIABLE WINDOW SIZE SELECTION

*These data association algorithms, based on the multidimensional assignment problem, range from sequential processing to many frames of data processed all at once! The data association problem for multisensor-multitarget tracking is formulated using a window sliding over the frames of data from the sensors. This discussion focuses on the work of Poore and Drummond and not on the formulations in Section III. Firm data association decisions for existing track are made at, say frame  $k$ , with the most recent frame being  $k+M$ . Decisions after frame  $k$  are soft decisions. Reports not in the confirmed tracks are used to initiate tracks over frames numbered  $k-I$  to  $k+M$ .*

The length of these windows varies from sequential processing, which corresponds precisely to  $I = 0$  and  $M = 1$ , to user defined large values of  $I$  and  $M$ . There is a marked change in performance over this range. As the two windows become

5 exceedingly long, there is little statistically significant information gained and indeed performance can degenerate slightly. Thus, one must manually find the correct window lengths for performance in a given scenario and the algorithms do not change for a changing environment. Thus we propose an

10 adaptive method for adjusting the window lengths. (The method has been highly successful for selecting the correct window length for multistep methods in ordinary differential equations.)

1. Compute the solution for different window lengths that

15 overlap and differ by one or two frames.

2. Compare the solution quality, i.e., the value of the objective function, for two adjacent window lengths.
- 20 3. If there is a predefined improvement in either direction, we then, for stability, repeat the exercise for a shorter or longer on the first try. If there is consistent information, we adjust the window size in the indicated



direction. This can be given a well defined mathematical formula in terms of the assignment problems of different dimensions.

5

10

#### X. SENSITIVITY ANALYSIS

For sequential processing in which the two-dimensional assignment algorithm is used, one can use the LP sensitivity analysis theory and easily obtain the corresponding answers.

15 For the multiframe processing, the optimal solution is not available; however, there are still two approaches one can use. First, the basic algorithm can perform the same sensitivity analysis at each stage (loop) of the algorithm as is done in the two-dimensional assignment problem since the

20 evaluation of function  $\Phi_{N-k+1}$  is equivalent to a two-dimensional assignment problem. Alternately, one could use an LP relaxation of the assignment problem and base the sensitivity on the resulting LP problem. We currently see this as an

important step in finding even better solutions to the assignment problem if so desired.

5

10

# XI. NEW AUCTION ALGORITHMS.

In this section we present a new auction algorithm for the two-dimensional assignment problem.

15       An important step in solving  $N$ -dimensional assignment problems for  $N \geq 3$  is finding the optimal solution of the two-dimensional assignment problem. In particular, we wish to solve the two-dimensional problem which includes the zero-index. Typically this problem can be thought of as trying to  
20 find either a minimum cost or maximum utility in assigning person to objects, tenants to apartments, or teachers to classrooms. We will follow the work of Bertsekas and call the first index set persons and the second index set objects. The

statement of this problem is given below (11.1) when the number of persons is  $m$  and the number of objects is  $n$ .

$$\text{Minimize} \quad \sum_{i=0}^m \sum_{j=0}^n c_{ij} x_{ij} \quad (11.1)$$

$$\text{Subject to} \quad \sum_{j=0}^n x_{ij} = 1 \text{ for all } i=1, \dots, m,$$

$$\sum_{i=0}^m x_{ij} = 1 \text{ for all } j=1, \dots, n.$$

There are a couple of assumptions which we will make about (11.1). First of all, we assume that  $c_{i0}$  and  $c_{0j}$  are well-defined and the corresponding variables  $x_{i0}$  and  $x_{0j}$  are free to be assigned. Second if a cost  $c_{ij}$  happens to be undefined, then the corresponding variable  $x_{ij}$  will be set to 0. In effect if  $c_{ij}$  is undefined, we simply remove this cost and variable from inclusion in the problem.

Notice that there are no constraints on the number of times person 0 and object 0 can be assigned. But notice that the first constraint requires that each non-zero person  $i$  must be assigned exactly once. Similarly, the second constraint forces each non-zero object  $j$  to be assigned exactly one time. Finally, the last constraint gives an integer solution, although we will see shortly that this constraint can be relaxed to admit any solution  $x_{ij} \geq 0$ . One reason for requiring that all of the costs of the form  $c_{i0}$  and  $c_{0j}$  be defined is so

that we are guaranteed a feasible solution exists for the given problem.

### XI.1. Relaxation of One Constraint

Relaxing the last constraint, we obtain the following  
5 problem:

$$\begin{aligned}
 &\text{Minimize} \quad \sum_{i=0}^m \sum_{j=0}^n c_{ij} x_{ij} + \sum_{j=0}^n u_j^2 \left| \sum_{i=0}^m x_{ij} - 1 \right| \\
 &\quad \equiv \sum_{i=0}^m \sum_{j=0}^n \left[ c_{ij} + u_j^2 \right] x_{ij} - \sum_{j=0}^n u_j^2 \\
 &\text{Subject to} \quad \sum_{j=0}^n x_{ij} = 1 \quad \text{for all } i=1, \dots, m \\
 &\quad \quad \quad x_{ij} \in \{0, 1\}.
 \end{aligned}$$

This lower bound is achieved and the second constraint in the original problem is satisfied if the following conditions  
10 are met:

$$\text{For } i=0, \quad x_{0j} = \begin{cases} 1, & \text{if } c_{0j} + u_j^2 \leq 0 \\ 0, & \text{otherwise} \end{cases},$$

$$\text{For } i \neq 0, \quad x_{ij} = \begin{cases} 1, & \text{if } j = \arg \min \{ c_{ik} + u_k^2 \mid k=0, 1, \dots, n \} \\ 0 & \text{otherwise} \end{cases}$$

All of  $j$ 's are assigned only one time.

The relaxation of the first constraint is analogous and would lead to similar results with  $i$  and  $j$  exchanged and the  
15 introduction of the multiplier  $u^1$ .

### XI.2. Relaxation of Both Constraints

This time we will relax both constraints and use duality theory to obtain a relationship between the multipliers  $u^1$  and  $u^2$ .

**Definition:** An assignment  $S$  and a multiplier set  $(u^1, u^2)$  are  
 5 said to satisfy  $\epsilon$ -complementary slackness ( $\epsilon$  - CS) if

$$\begin{aligned} c_{ij} + u_i^1 + u_j^2 &= 0 \text{ for all } (i, j) \in S, \\ c_{ij} + u_i^1 + u_j^2 &\geq -\epsilon \text{ for all } (i, j) \in A. \end{aligned}$$

#### Forward Auction Algorithm

- (1) Select any unassigned person  $i$
- (2) Determine the following quantities:

$$j_i = \arg \min \left\{ c_{ik} + u_k^2 \mid k \in A(i) \right\},$$

$$v_i = c_{ij_i} + u_{j_i}^2,$$

$$w_i = \min \left\{ c_{ik} + u_k^2 \mid k \in A(i), k \neq j_i \right\}.$$

10

In the selection of  $j_i$  above, if a tie occurs between 0 and any non-zero index  $k$ , then select  $j_i$  as  $k$ . Otherwise, if there is a tie between two or more non-zero indices, the choice of  $j_i$  is arbitrary.

15 Also if  $A(i)$  consists of only one element, then set  $w_i = \infty$ .

- (3) Update the multipliers and the assignment:

If  $j_i = 0$ , then

(a) Add  $(i, 0)$  to the assignment.

(b) Update  $u_i^1 := -c_{i0}$ .

If  $j_i \neq 0$ , then

(a) Add  $(i, j_i)$  to the assignment.

5 (b) Remove  $(i', j_i)$  from the assignment if  $j_i$  was previously assigned.

(c) Update  $u_{j_i}^2 := u_{j_i}^2 + (w_i - v_i) + \epsilon = w_i - c_{ij_i} + \epsilon$ .

(d) Update  $u_i^1 := -\left(c_{ij_i} + u_{j_i}^2\right) = -w_i - \epsilon$ .

10

#### Reverse Auction Algorithm.

(1) Select any unassigned object  $j$ , such that  $c_{0j} + u_j^2 > 0$ .

(2) Determine the following quantities:

$$i_j = \arg \min \left\{ c_{jk} + u_k^1 \mid k \in B(j) \right\},$$

$$\beta_j = c_{ij_j} + u_{i_j}^1,$$

$$\gamma_j = \min \left\{ c_{kj} + u_k^1 \mid k \in B(j), k \neq i_j \right\}.$$

15

In the selection of  $i_j$  above, if a tie occurs between 0 and any non-zero index  $k$ , then select  $j_i$  as  $k$ . Otherwise, if there is a tie between two or more non-zero indices, the choice of  $j_i$  is arbitrary.

Also if  $B(j)$  consists of only one element, then set  $\gamma_j = \infty$ .

(3) Update the multipliers and the assignment:

If  $i_j = 0$ , then

- 5           (a) Add  $(0, j)$  to the assignment.  
              (b) Update  $u_j^2 := -c_{0j}$ .

If  $i_j \neq 0$ , then

- (a) Add  $(i_j, j)$  to the assignment.  
 10           (b) Remove  $(i_j, j)$  from the assignment if  $(i_j, j)$  was  
              previously assigned.  
              (c) Update  $u_{i_j}^1 := u_{i_j}^1 + (\gamma_j - \beta_j) + \epsilon = \gamma_j - c_{i_j j} + \epsilon$ .  
              (d) Update  $u_j^2 := -(c_{i_j j} + u_{i_j}^1) = -\gamma_j - \epsilon$ .

#### Combined Forward/Reverse Auction Algorithm

- 15           1. Assume that  $u^2$  is given as an arbitrary multiplier.  
              2. Adjust the value of  $u^2$  for each object  $j$  as follows:  
                     If  $c_{0j} + u_j^2 < 0$ , then set  $u_j^2 := -c_{0j}$ .  
              3. Run iterations of the Forward Auction Algorithm  
                     until all persons become assigned.  
 20           4. Run iterations of the Reverse Auction Algorithm  
                     until all of the objects become assigned.

Note at the completion of the Forward auction step we have the following conditions satisfied:

- $c_{0j} + u_j^2 \geq 0$  for all objects  $j$ .
- $c_{ij} + u_i^1 + u_j^2 = 0$  for all  $(i, j) \in S$ .
- $c_{ij} + u_j^2 \leq \min_k \{c_{ik} + u_k^2\} + \epsilon$  for all  $(i, j) \in S$ .

Thus we can prove the following proposition.

- 5 Proposition: If we assume that  $c_{0j} + u_j^2 \geq 0$  at the start of the Forward Auction Algorithm and all of the persons are assigned via a forward step, then we have:

$$c_{ij} + u_i^1 + u_j^2 \geq -\epsilon \quad \text{for all } (i, j) \in A.$$

$$c_{ij} + u_i^1 + u_j^2 = 0 \quad \text{for all } (i, j) \in S.$$

10 
$$c_{ij} + u_j^2 \leq \min_k \{c_{ik} + u_k^2\} + \epsilon \quad \text{for all } (i, j) \in S.$$

### Optimality of the Algorithm

Theorem:  $\epsilon$ -CS preserved during every forward and reverse iteration.

- 15 Theorem: If a feasible solution exists, then the resulting solution is with  $m\epsilon$  of being optimal for the Combined Forward Reverse Algorithm.

## XII. SOME CONCLUDING COMMENTS

- 20 Although the algorithm appears to be serial in nature, its primary computational requirements are almost entirely parallelizable. Thus parallelization is planned.



Step 2 is the computational part of the algorithm. Evaluating  $\Phi_{N-k+1}$  and computing a subgradient use in the search procedure requires 99% of the computing time in the algorithm. This part uses two-dimensional assignment algorithms, a search  
5 over a large number of indices, and a nonsmooth optimization algorithm. It is the second part (the search) that consumes 99% of the computational time and this is almost entirely parallelizable. Indeed, there are two-dimensional assignment solvers that are highly parallelizable. Thus, we need but  
10 parallelize the nonsmooth optimization solver to have a reasonably complete parallelization.

If a sensitivity analysis is desired or if one is interested in computing several near-optimal solutions, a parallel processor with a few powerful processors and good  
15 communication such as on the Intel Paragon would be most beneficial.

The foregoing discussion of the invention has been presented for purposes of illustration and description. Further, the description is not intended to limit the  
20 invention to the form disclosed herein. Consequently, variation and modification commensurate with the above teachings, within the skill and knowledge of the relevant art, are within the scope of the present invention. The embodiment

described hereinabove is further intended to explain the best mode presently known of practicing the invention and to enable others skilled in the art to utilize the invention as such, or in other embodiments, and with the various modifications  
5 required by their particular application or use of the invention. It is intended that the appended claims be construed to include alternative embodiments of the invention to the extent permitted by the prior art.

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2080  
2081  
2082  
2083  
2084  
2085  
2086  
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2099  
2100  
2101  
2102  
2103  
2104  
2105  
2106  
2107  
2108  
2109  
2110  
2111  
2112  
2113  
2114  
2115  
2116  
2117  
2118  
2119  
2120  
2121  
2122  
2123  
2124  
2125  
2126  
2127  
2128  
2129  
2130  
2131  
2132  
2133  
2134  
2135  
2136  
2137  
2138  
2139  
2140  
2141  
2142  
2143  
2144  
2145  
2146  
2147  
2148  
2149  
2150  
2151  
2152  
2153  
2154  
2155  
2156  
2157  
2158  
2159  
2160  
2161  
2162  
2163  
2164  
2165  
2166  
2167  
2168  
2169  
2170  
2171  
2172  
2173  
2174  
2175  
2176  
2177  
2178  
2179  
2180  
2181  
2182  
2183  
2184  
2185  
2186  
2187  
2188  
2189  
2190  
2191  
2192  
2193  
2194  
2195  
2196  
2197  
2198  
2199  
2200  
2201  
2202  
2203  
2204  
2205  
2206  
2207  
2208  
2209  
2210  
2211  
2212  
2213  
2214  
2215  
2216  
2217  
2218  
2219